A Wearable Real-Time System for Physical Activity Recognition and Fall Detection

A Thesis Submitted to the College of
Graduate Studies and Research
In Partial Fulfillment of the Requirements
For the Degree of Master of Science
In the Department of Electrical and Computer Engineering
University of Saskatchewan
Saskatoon

By

Xiuxin Yang

PERMISSION TO USE

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Electrical and Computer Engineering

57 Campus Drive

University of Saskatchewan

Saskatoon, Saskatchewan, Canada

S7N 5A9

ACKNOWLEDGMENTS

First and foremost I offer my sincerest gratitude to my supervisors Dr. Anh Dinh and Dr. Li Chen, for their supervision, advice, and guidance from the very early stage of this research as well as giving me extraordinary experiences throughout the work, using their precious time to read this thesis and give their critical comments about it. The completion of this thesis would not have been possible without Dr. Dinh and Dr. Chen's exceptional supervision and everlasting support.

I would like to show my gratitude to FANFARE group, especially Dr. Daniel Teng, who gave me valuable discussion and advice. I gratefully thank Mr. Zhongkai Chen for his assistance in implementation with the Sun SPOT. I also appreciate all the members of VLSI lab. Funding from NSERC under Strategic Project Grant number STPGP 350545 is gratefully appreciated.

I wish to thank my parents, Yang Shengqian, Bai Shuqiong, and my brother Yang Xiuquan for their continuous support and encouragement throughout my studies.

ABSTRACT

This thesis work designs and implements a wearable system to recognize physical activities and detect fall in real time. Recognizing people's physical activity has a broad range of applications. These include helping people maintaining their energy balance by developing health assessment and intervention tools, investigating the links between common diseases and levels of physical activity, and providing feedback to motivate individuals to exercise. In addition, fall detection has become a hot research topic due to the increasing population over 65 throughout the world, as well as the serious effects and problems caused by fall.

In this work, the Sun SPOT wireless sensor system is used as the hardware platform to recognize physical activity and detect fall. The sensors with tri-axis accelerometers are used to collect acceleration data, which are further processed and extracted with useful information. The evaluation results from various algorithms indicate that Naive Bayes algorithm works better than other popular algorithms both in accuracy and implementation in this particular application.

This wearable system works in two modes: indoor and outdoor, depending on user's demand. Naive Bayes classifier is successfully implemented in the Sun SPOT sensor. The results of evaluating sampling rate denote that 20 Hz is an optimal sampling frequency in this application. If only one sensor is available to recognize physical activity, the best location is attaching it to the thigh. If two sensors are available, the combination at the left thigh and the right thigh is the best option, 90.52% overall accuracy in the experiment.

For fall detection, a master sensor is attached to the chest, and a slave sensor is attached to the thigh to collect acceleration data. The results show that all falls are successfully detected. Forward, backward, leftward and rightward falls have been distinguished from standing and walking using the fall detection algorithm. Normal physical activities are not misclassified as fall, and there is no false alarm in fall detection while the user is wearing the system in daily life.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ANN	Artificial Neural Network
EE	Energy Expenditure
GSM	Global System for Mobile Communications
GUI	Graphic User Interface
NB	Naive Bayes
PA	Physical Activity
SMS	Short Message Service
Sun SPOT	Sun Small Programmable Object Technology
SVM	Support Vector Machines
WEKA	Waikato Environment for Knowledge Analysis

Chapter 1 Introduction

In this chapter, the motivation of physical activity recognition as well as fall detection is introduced. Then the characteristics of fall are discussed before an effective fall detection approach is proposed. Five main challenges in the thesis work are to be discussed. Finally the objective and structure of this thesis work are given.

1.1 Motivation of physical activity recognition

Physical activity has been defined as "any bodily movement produced by skeletal muscles that results in energy expenditure" [1]. The energy expenditure can be measured in kilocalories. Physical activity in daily life can be categorized into sitting, standing, lying, walking, walking up the stairs, running etc. Physical activity recognition involves the use of modern technology to automatically recognize different activities and recording relevant information [2].

There are numerous applications of physical activity recognition. Automatic recognizing physical activity helps people maintain their energy balance and stay physically fit and healthy by developing health assessment and intervention tools [3]. It is valuable to investigate the links between common diseases and levels of physical activity. Cardiovascular disease and diabetes mellitus and depression are reported to be related to physical inactivity [2]. Although self-reporting is employed to quantify physical activity, fully automated activity classification which is more reliable and easy to implement is considered as the optimal solution.

Activity recognition systems are also used to provide feedback to motivate individuals to adhere to daily or weekly physical activity targets [2]. The information provided by the system can evaluate the levels of physical activity and analyze the reason why people choose to exercise. Accurately recognizing physical activity helps to improve the treatment and differential diagnosis of neurological, degenerative and respiratory disorders. Automatic activity classification systems have been used in patients with Parkinson's disease. Researchers acquire the vertical linear acceleration of the left shank by an ankle-mounted sensor array that transmits data wirelessly to a pocket PC [4]. The systems are used to access the physical activity levels in patients with multiple sclerosis, osteoarthritis and chronic pulmonary disease. They are also used to assess effectiveness of medical treatments [2]. Functional parameters, such as gait speed and energy expenditure can be also estimated by physical activity information.

1.2 Motivation of fall detection

Fall detection has become a hot research topic due to the increasing population over 65 throughout the world, as well as the serious effects and problems caused by fall. "A fall is often defined as a sudden and unintentional change in position resulting in an individual landing at a lower level such as on an object, the floor, or the ground, with or without injury" [5]. Since the population of the elderly increases, the number of falls and fall-related injuries are expected to increase proportionally. Unless effective fall prevention approaches are proposed and implemented, the personal and societal costs of falls will steadily increase with the aging population [6].

Population aging is a common feature of developed countries. With the development of society, better living conditions as well as more advanced medical technologies, never before in history have people lived so long. In seven developed countries (Canada, France, Germany, Italy, Japan, the United Kingdom, and the United States), the average percentage of the population live over 65 is expected to grow from 15% to 27% in the next 50 years. When more and more people over 65, their physical and mental health tends to deteriorate. So they need more nursing care and other healthcare services [7].

In Canada, people born in the 1960 can expect to live 20 years longer than Canadians who were born in the 1900. The birth rates also have declined, so that the proportion of the population over 65 increases. By the year 2031, approximately 20% of Canada's population, i.e., one in five will be seniors. Figure 1.1 shows Canada's aging population from 1900 to 2030, according to Statics Canada (as cited in [8]). Furthermore, researchers assort seniors into the young-old, the middle-old and the old-old, and by 2031 the old-old are estimated to make up 45% of the elderly [8], which means more health care are needed. With the growing problem of rising health care costs, many people worry that the increasing burden is put on the Canadian health care system.

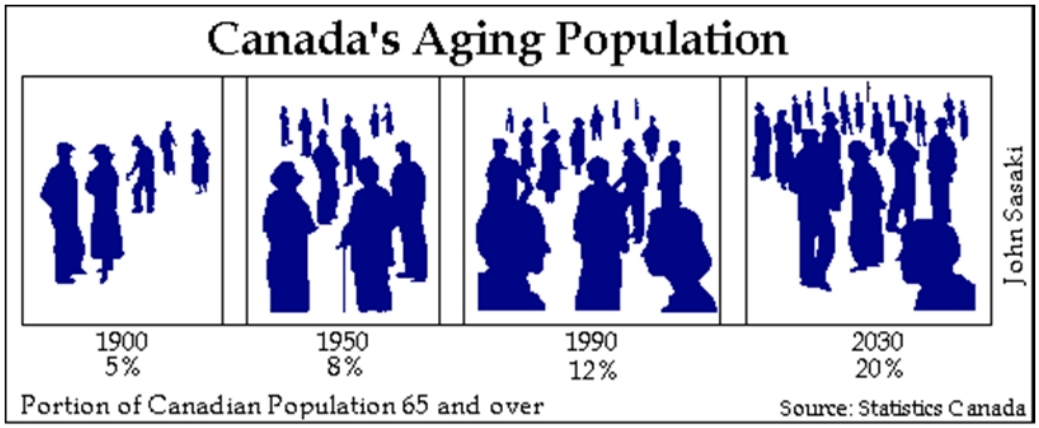

Figure 1.1 Canada's aging population [8]

The effects and problems of falls are serious. "Falls are the second leading cause, after motor vehicle collisions, of injury-related hospitalizations for all ages, accounting for 29% of injury admissions [9]." Falls cause 85% of injury hospitalizations in the age group of 65 and over [10]. More seriously, falls cause more than 90% of all hip fractures and 20% die within a year of the fracture [11]. Even without an injury, falls can lead to a loss in confidence and curtailment of activities, which can cause a decline in health and function and contribute to future falls with more serious outcomes [12]. Figure 1.2 shows the Canadian old people age over 65, self-reported falls resulting in injury, by type of injury [5]. From the Figure 1.2, 37% injuries happen at the hip, thigh, knee, lower leg, ankle as well as foot; 17% injuries happen at the wrist and hand. Every year, one in three Canadian seniors will fall at least once. Hip fractures are the most common type of fall

injury among seniors, and about 20 percent of injury-related deaths among seniors can be traced back to a fall [13].

For the elderly people themselves, hip fractures and head injuries are serious effects which may cause accidental death. For government and medical organizations, an effective approach to avoid and prevent fall among the elderly is able to reduce expenditure and decrease the pressure of shortage of healthcare personnel. "A 20% reduction in falls would translate to an estimated 7,500 fewer hospitalizations and 1,800 fewer permanently disabled seniors. The overall national savings could amount to $138 million annually", according to SMARTRISK [5].

Since there are many serious effects caused by falls, an efficient system that is designed to detect falls and prevent falls will benefit the society. In order to reduce falls, the most critical thing is to detect falls accurately before triggering a certain protection system so that the elderly who are wearing the system can be protected.

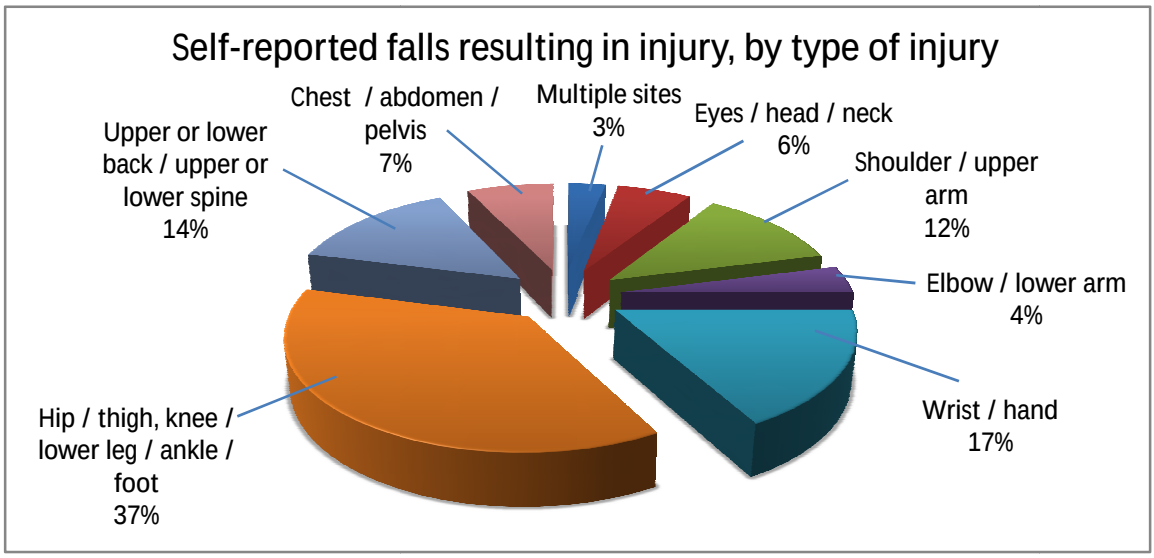

Figure 1.2 Self-reported falls resulting in injury, by type of injury, age 65+, Canada, 2002/03 [5]

1.3 Characteristics of fall

Based on the direction, falls can be sorted as: forward, backward, leftward and rightward fall. Lord et al. [14] found that 82% of falls occurred from the standing height, which means detecting falls from daily physical activities such as standing and walking is the most important issue. Previous research has shown that 60% of falls in elderly people are in forward direction [15] , while backward fall cannot be ignored due to its related serious injuries [16]. Leftward and rightward falls also potentially fracture elderly people's hip with serious consequences when they happen [17].

Based on the falling height, falls can be divided into three types: fall from lying; fall from sitting and fall from standing or walking. Since fall from lower height causes less severe effect to the elderly, we only discuss fall from standing or walking, which is the leading cause of bone fracture and related injuries.

Some characteristics of fall are listed below [18]:

(a). A fall lasts 1 to 2 seconds, consisting of several sub-actions.

(b). A fall starts from a standing still position, and ends when the body completely contacts the ground.

(c). A person falls roughly in one direction.

(d). Within the falling period, the head will fall in a free fall manner.

Although people in all ages fall at one time or another in their lives, the elderly are particularly at risk for falls that also have more serious effects. The factors increasing the risk of falling include: improper balance, declined strength of muscle and bone, impaired vision or hearing, unsafe conditions in and around home, gait inconsistency [13].

In terms of defense against such possible falls accompanied by aging, increasing the exercise of the elderly to an appropriate level will effectively maintain their physical strength and decrease the injuries caused by falls. However, except for the exercise performed by the elderly themselves, there are other effective approaches to detect and prevent falls. Three fall detection approaches are proposed in previous research, including wearable-based, camera-based and ambience-based [18]. The wearable-based fall detection is widely used due to the flexibility of accelerometers or gyroscopes in measuring people's movement. The details of this approach will be discussed in Chapter

2. One example is that Tamura et al. [19] employs one tri-axis accelerometer and one gyroscope to detect movement so that the wearable airbag is inflated once a fall occurs.

1.4 Challenges

For physical activity recognition and fall detection, we use wearable sensors with tri-axis accelerometer to collect movement information from a human body. There are five main challenges.

1) The first challenge is to determine how many sensors will be appropriate to collect data. Theoretically, more sensors attached to different locations on the body will increase the detection accuracy, because they provide more attributes to build and test the classification model. However, there is a trade-off in the use of more sensors. First, the number of attributes increases proportionally once the number of sensor increases. Providing each tri-axis accelerometer has 3 attributes, there are 15 attributes when 5 sensors are attached to the body, which will make the classification algorithm highly complicated. Especially in regular embedded system, the computational capacity and on-chip memory cannot satisfy the complex algorithm demands. For instance, the simple Naive Bayes classifier must perform 15 exponentiation operations to produce 1 classification result for 15 attributes, which makes the ARM9 microprocessor perform quite slowly. Long time delay of each calculation prohibits the real-time classification. In addition, more sensors also make wireless communication among sensors more complicated, consume more power and make the system less comfortable to use. Therefore, we do experiments to find out the optimal number of sensors with an acceptable accuracy.
2) Another challenge is the sensor placement on a human body. Although different locations on a human body can be used to attach sensors, such as at the head, chest, wrist, waist, back, hip, thigh, calf and even foot, the movement information the sensors provide varies. Thus some experiments must be performed to evaluate the classifier performance on different sensor locations.

3) The third challenge is based on the variety of people's physical activity as well as fall. People have different physical activities, such as walking style and sitting postures. Fall also varies in direction and speed. In order to denote the difference, we simulated falls in various directions and record the time consumed, as shown in Table 1-1 the timing of different falls, where all the falls are simulated from a static standing position down to 34 cm height mattress. Fall time is defined as the time interval from standing position to people's body completely contacting the mattress. From Table 1-1, the fall time in different direction varies, with the average fall time less than 2000ms.These data are collected from one subject; data may vary among different people. Due to the variety mentioned above, the traditional threshold classifier cannot adapt to different individuals. Artificial intelligence classifier, such as neural network, support vector machine as well as Naive Bayes algorithm, must be used to build classification system.

Table 1-1 Timing of simulated falls in different direction from a static standing height to 34 cm height mattress

	Forward fall (ms)	**Backward fall (ms)**	**Leftward fall (ms)**	**Rightward fall (ms)**
1	1640	1830	1700	1750
2	1510	2060	1560	1700
3	1260	1690	1440	1450
4	1370	1760	1560	1560
5	1190	1630	1620	1500
Average	1394	1794	1576	1592

4) The fourth challenge is the fall detection accuracy. The ideal system will accurately detect every fall, and should not misclassify normal physical activity as fall so that no false alarm is generated. This is challenging because some intensive physical activity, such as walking fast could be misclassified as fall.
5) The last challenge is how fast the system can successfully detect a fall. The objective of fall detection system is to generate a signal to trigger protection equipment and finally prevent fall and related injuries to the elderly, one such a device is a wearable airbag protection system. If the system detects fall fast enough, there will be adequate time to switch on the protection equipment (e.g., inflate the airbag). Simple classifier and appropriate data processing technique may increase the detection speed.

Figure 1.3 shows the timing of falling down. This also describes the relationship between detection system and protection system. A testing subject starts in a standing still position, and ends when the subject falls down to the ground. T1 is the time when subject starts to fall. T2 is the time when the detection system detects subject's falling down. Based on the acceleration data collected at T2, the detection system starts to calculate. Rectangle 2 is the duration of calculation. At T3 the detection system makes a decision that the subject will fall and trigger the protection system. At T4 the protection system takes action (e.g., completely inflate the airbag) then prevents a fall. Rectangle 3 is the mechanical response time. T5 is the time when the body completely contacts the ground.

The duration between T1 and T5 is falling time (i.e., Rectangle 1), which is less than 2000ms. The duration between T2 and T3 (i.e., Rectangle 2) is the calculation time of detection system. The duration between T3 and T4 (i.e., Rectangle 3) is the response time of protection system. In one previous research, Tamura et al. designed a wearable airbag to prevent fall injuries, where the airbag inflation average time was 121ms [19]. In modern auto industry, the airbag can be fully inflated within approximately 60-80ms after the first moment of vehicle contact. In order to successfully prevent a fall, the following equation must be subject to:

$$\text{Rectangle 2} + \text{Rectangle 3} < \text{Rectangle 1} \tag{1.1}$$

In this thesis work, we only discuss the fall detection rather than protection, so a fast algorithm with easy implementation must be selected to shorten the duration between T2 and T3 (Rectangle 2) to make the system more sensitive and accurate.

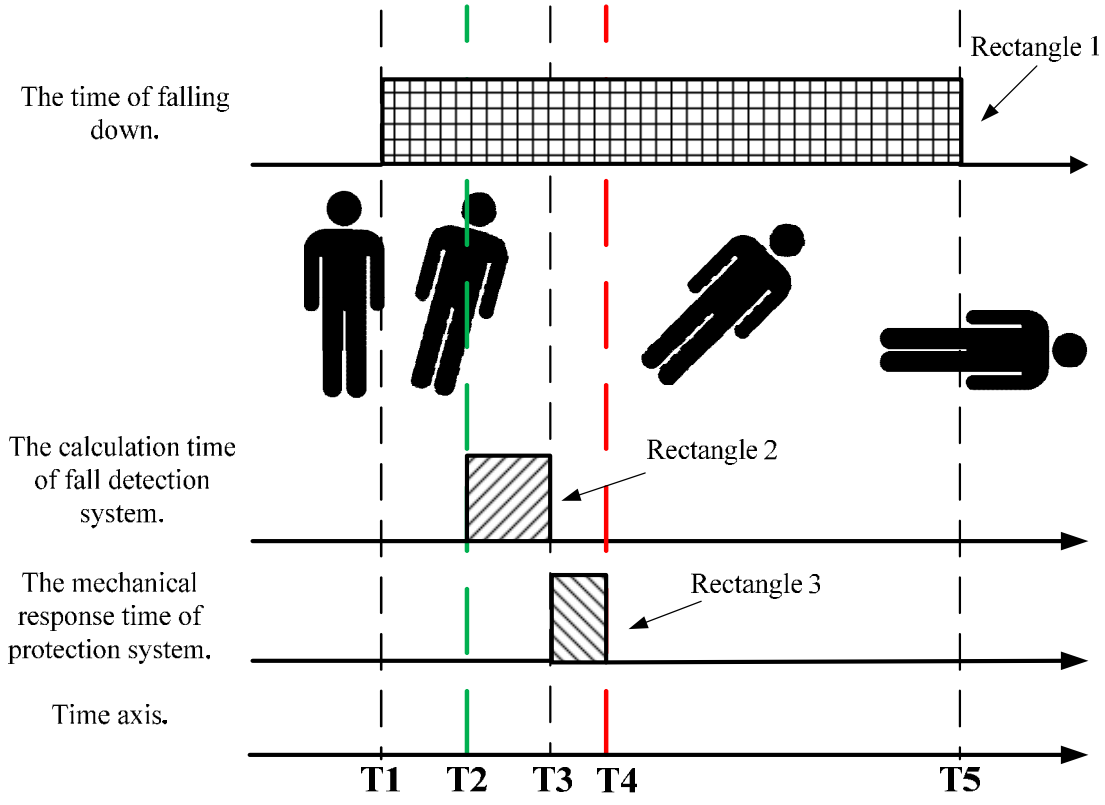

Figure 1.3 Timing of falling down

1.5 Objectives of thesis work

The thesis work intends to design and implement a system that can recognize physical activity and detect fall. The real-time system is designed for wearable application. The basic requirements should be satisfied: (i) The system must be small, light weight and easy to wear, (ii) The system must be able to recognize daily physical activity and detect fall in real time with relatively high accuracy. The thesis work will be considered a success if:

- **Recognize physical activity:** The daily physical activity includes sitting, standing, lying, walking and walking up the stairs. Since the system is originally design for

the elderly, the running activity is excluded. The recognition accuracy should be relatively high.

- **Detect fall:** The system should detect falls in four directions: forward, backward, leftward and rightward. Once a fall happens, it is able to generate an alert signal immediately and indicate which direction the fall is. Meanwhile, the normal physical activity won't be misclassified as fall.
- **Evaluate different classifiers:** Performance of popular artificial intelligence classifiers will be evaluated, so that the optimal one is applied.
- **Evaluate the sensor placement on human body:** Under exactly the same algorithm and hardware, the senor is attached to different locations on human body so that the performance is compared.
- **Evaluate the sensor numbers on human body:** One sensor and two, or even more than two sensors are used to collect acceleration data respectively. The performance is also compared.
- **Evaluate different sampling rate:** Using exactly the same hardware, the system works in different sampling rates, then compare the detection results.

1.6 Structure of the thesis

Chapter 1 introduces the motivations of this thesis work, the characteristics of fall and physical activity, as well as the objectives of thesis work. Chapter 2 is the literature review, including previous work in this area. Chapter 3 describes the system in both hardware and software, e.g., the sensor development platform as well as classification strategy. Chapter 4 compares popular artificial intelligence algorithms in WEKA; finally gives the details of how the real-time system recognizes physical activity and detects fall. Chapter 5 describes how Naive Bayes classifier is implemented in the Sun SPOT sensors. Chapter 6 and 7 present classification results, discussion and conclusion.

Chapter 2 Literature Review

This chapter reviews some previous literatures related to physical activity recognition and fall detection. First, different approaches are reviewed and compared, since wearable sensor with accelerometers has been proved to have more advantages. Second, the sensor number and their placement are also discussed because they are important factors for improving detection accuracy. Finally, another critical factor is the classification algorithm, which not only determines the detection accuracy, but also limits the implementation (e.g., operation speed, hardware requirement, etc.).

2.1 Different approaches

It is difficult to accurately measure all aspects of physical activity due to people's complex activity nature and diversified individual behaviors. The current available measuring techniques can be grouped into five categories: behavioral observation, self-report, physiological markers (e.g., heart rate, body temperature, ventilation), motion sensors (e.g., pedometers, accelerometers), and indirect calorimetry, according to Plasqui and Westerterp [20]. Previous researches have shown that physical activity are recognized using body-mounted sensors [2] or by accelerometry [21]. Wearable accelerometers and gyroscopes provide people's movement information, which will be further processed by filters or other classifiers.

For fall detection, there are three different approaches according to how fall is detected: wearable device, ambience device and camera-based, as shown in Figure 2.1.

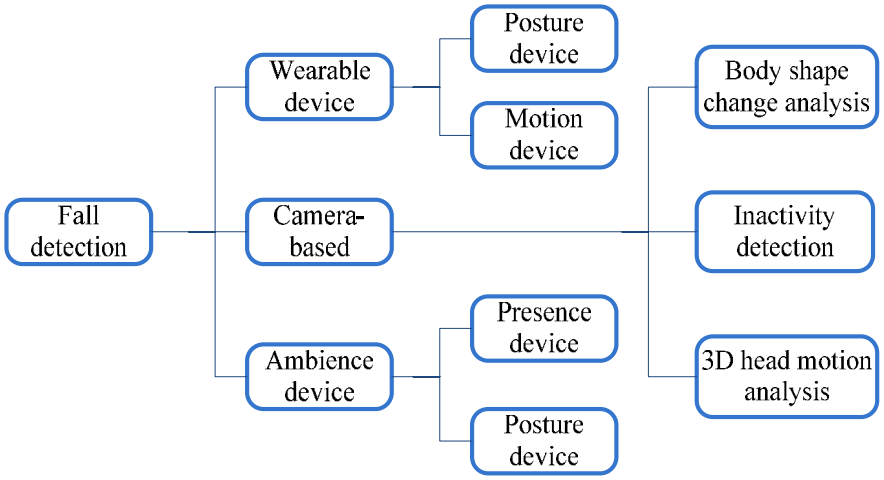

Figure 2.1 The hierarchy of approaches and classes of fall detection methods [18]

According to this hierarchy, three approaches have advantages and disadvantages, as show in Table 2-1.

Table 2-1 Comparison of three approaches of fall detection [18]

	Definition	**Typical example**	**Advantages**	**Disadvantages**
Wearable device	User wears some devices with embedded sensors to detect posture or motion	Accele-rometer, gyroscope, mercury	Cheap; easy to set up	High rate of false alarm; the sensor is intrusive
Camera-based	Visualization with camera	Single or multiple cameras	Monitor multiple events simultaneously; less intrusive; the recorded video for remote and post verification	The accuracy is very sensitive to lighting condition, dim light in the night with poor performance; only indoor application; possible to divulge private matters
Ambience device	Use multiple installed sensors to collect the data when users are closed to them	Pressure sensor to obtain user's location	Cheap and non-intrusive	We cannot discern if pressure is from the user's weight; They cannot be visually verified

The final objective of fall detection is to generate a fall alarm and trigger protection system (i.e., an airbag or other means) so that people are properly protected. Some previous research detect fall based on confirming people's lying position on the ground. While such a detecting strategy maybe have high accuracy, it could do nothing in preventing falls or protecting people. Thus actual system should accurately detect potential falls and trigger protection system before they cause serious injuries to people.

2.2 Sensor numbers and placement

In terms of detection accuracy, the number of sensors and their placement on the body are two critical factors. The placement of sensors depends on the motivation as well as the algorithm. In some applications, only one sensor is placed at a certain location and used to collect movement information. However in other applications, more than two sensors are used.

Previous research used one, two, three or five sensors to collect acceleration data, either wired or wireless. More sensors might increase accuracy, but the shortcoming is that wireless communications among sensors makes the system more complicated and less reliable. In addition, more sensors considerably increase the computation time of the microprocessor hence reduce performance of the classifier. Also wearing more sensors is uncomfortable for the users.

Typical laboratory and clinical studies done using accelerometers are summarized in Table 2-2.

From Table 2-2, there is variability in sensor numbers, placement and activities; therefore it is not possible to directly compare classification accuracies between different studies.

Table 2-2 Laboratory and clinical studies done using accelerometers [21]

	Author	Placement	Motivation	Algorithm
1 sensor	Lee et al. (2003, as cited in [21])	1 back	Physical activity: 5 static activities (stand, sit, lower head, sit down & lean against, lie supine & prone) and 4 dynamic activities (walk, run, up/down stairs)	Thresholds to DC (static), thresholds to AC analysis (dynamics), video analysis
2 sensors	NiScanaill et al. (2006, as cited in [21])	1 trunk, 1 thigh	Remote sensor for home care: sit, stand, lie, walk	Means, thresholds, SMS message on GSM network
3 sensors	Hester et al. (2006, as cited in [21])	1 wrist, 1 ankle, 1 walking stick	Stroke patients: motor tasks at home-assessment of mobility assistive devices (cane) (accelerometer + gyroscope)	Dominant frequencies, energy aspects, cross-correlations, auto-covariance's, NN, threshold, wireless transmission
4 sensors	Noury et al. (2004, as cited in [21])	1 chest, 1 wrist, 1 thigh, 1 ankle	PA young and old: walking, postural transitions (accelerometers + magnetometer), orientation angles	Frequency spectrum analysis
5 sensors	Bao et al. (2004, as cited in [21])	1 wrist, 1 waist, 1 upper arm, 1 thigh, 1 leg	Walking, sit and relax, stand, watch television, run, stretch, scrubbing, fold laundry, brush teeth, ride elevator, walk + carry, read, cycle, climb stairs, vacuuming, lie down, strength training, etc.	Mean, energy, frequency domain entropy, correlation of acceleration data, classifiers: C4.5 decision tree, decision table, Naive Bayes classifier, instance based learning (IBL)

2.3 Classification algorithm

Once the raw acceleration data are collected and processed, they are imported to a classification algorithm and finally generate the results. An appropriate classification algorithm not only improves the classification accuracy, but also simplifies the

implementation, which makes it possible to realize real-time performance. The degree of complexity of different classification algorithms varies from threshold-based schemes to more advantages algorithms, e.g., artificial neural networks or hidden Markov models [2].

The classification algorithms are categorized into two classes: supervised and unsupervised. Most algorithms in previous literatures belong to supervised machine learning techniques. Table 2-3(a) (b) and (c) show 11 algorithms, related brief description, advantages, disadvantages and related literatures.

Table 2-3(a) Typical machine learning techniques and literatures [2]

Machine learning technique	Brief description	Advantages	Disadvantages	Literature
Threshold-based classification	Differentiate between static postures, identify postural transitions, differentiate static postures and dynamic postures, detect falls	Easy to implement, fast execution	Device needs calibration, fixed threshold is hard to adapt to different individuals	Nyan et al. (2006)
Hierarchical methods	Binary decision structure (decision tree with two child nodes)	Fast execution, suitable to real-time applications; graphically represented, easy to understand	It takes long time to develop	Ermes et al. (2008)
Decision trees	Being constructed automatically and created a compact set of rules from root to leaf	Fast execution; faster to develop and less user intervention than Hierarchical methods; graphically represented, easy to understand	Over-sensitivity to the training set, to irrelevant attributes and to noise	Bao and Intille (2004); Ermes et al. (2008)

Table 2-3(b) Typical machine learning techniques and literatures [2]

Machine learning technique	Brief description	Advantages	Disadvantages	Literature
k-nearest neighbor	Calculate the distance of unknown point to known points; classification is determined by the majority of the k-nearest neighbors	Developed rapidly, highly versatile, classify a large range of different activities	Slower on-line execution than decision tree	Preece et al. (2008)
Artificial neural networks	A flexible mathematical function configured to represent complex relationships between inputs and outputs	Flexible, classify a large range of different activities, accurate results	Slow to train and test, difficult to implement some types of networks	Ermes et al. (2008)
Support vector machines	Based on finding optimal separating decision hyperplanes between classes with the maximum margin between patterns of each class	Powerful and popular, possible to work reliably with difficult and noisy classification datasets	Very slow to train with large datasets and difficult to set their kernel type and kernel parameters	Zhang et al. (2006)
Naive Bayes and Gaussian mixture models	Based on the estimated conditional probabilities or likelihoods of the signal patterns available from each activity class	Simple to develop, rapid execution, high accuracy	Based on the weak assumption of feature independence	Wu et al. (2007)

Table 2-3(c) Typical machine learning techniques and literatures [2]

Machine learning technique	Brief description	Advantages	Disadvantages	Literature
Fuzzy logic	Derived from fuzzy set theory. It allows mapping from a set of inputs to one or more outputs via a set of if-then statements called rules	A small number of previous studies demonstrate good classification accuracies	Difficulties in construction of appropriate membership functions	Boissy et al. (2007)
Markov models	Graphical models containing information on the probability of transition between different activity states	It works well as a single classifier and for improving the performance of other classifiers	NA	Ward et al. (2006)
Combining classifiers	Combine the outputs of different classifiers to improve the performance, e.g., majority voting, stacked generalization or boosting	Provide complementary decisions and improve the overall accuracy	Consume more computational resources and memory	Lester et al. (2005, 2006)
Unsupervised learning	For analysis and interpretation of data without the need of labels, identify clusters of related patterns in the feature space	Provide insight into the structure of activity data within the feature space	Always require some input from the user	Nguyen et al. (2007)

For a specific activity recognition problem, different classifiers can be evaluated in order to maximize the performance, such as accuracy and operation speed. However,

Preece et al [2] present that “there is no classifier which performs optimally for a given activity classification problem”. Each classifier has advantages and disadvantages. While selecting classifier, more than one factor should be considered, depending on the actual requirement. Accuracy, easy of development and speed of real-time execution are most critical factors.

2.4 Summary

Three aspects of related research have been reviewed. For the detection approach, the wearable device overtakes the camera-based as well as ambience device, because it provides users free mobility, suitable for indoor and outdoor applications. Previous researchers use various numbers of sensors to collect acceleration data, but it is difficult to simply compare results in different systems, due to the variety of sensor placement as well as classifier algorithm. Therefore, before a system is physically implemented, we should collect acceleration data from sensors attached to different locations and compare classification algorithms, based on which an optimal solution will be proposed.

A wearable real-time system is designed to meet the requirement both in outdoor and indoor applications. Additionally, there should be a suitable hardware development platform, which not only has internal tri-axis accelerometers, but also has powerful microprocessor to handle all tasks.

Chapter 3 System Overview

In this chapter, the hardware components are introduced. The Sun SPOT wireless sensor is used as the hardware development platform, which collects acceleration data and accomplishes calculation operation. Based on the hardware platform, two working modes are designed for indoor and outdoor application respectively. Then the software component is also discussed, including the training and testing strategy.

3.1 Hardware components

The hardware consists of two Sun SPOT wireless sensors and a laptop. There are two application modes: wireless sensors with laptop computer and wireless sensors without laptop computer.

3.1.1 The Sun SPOT wireless sensor

The Sun SPOT (Sun Small Programmable Object Technology, shown in Figure 3.1) is a small, wireless, battery-powered Java programmable embedded device designed for flexibility. The Sun SPOT sensor is primarily made up of a sensor board and a processor board. The basic unit includes accelerometer, temperature and light sensors, radio transmitter, eight multicolored LEDs, 2 push-button control switches, 5 digital I/O pins, 6 analog inputs, 4 digital outputs, and a rechargeable battery [22]. An embedded

ARM920T microprocessor is used to coordinate the tasks it performs. With the wireless module, the sensors can communicate among themselves wirelessly to build a wireless sensor network. In one set of the Sun SPOT kit, there are two sensors and a receiver which connects to a PC.

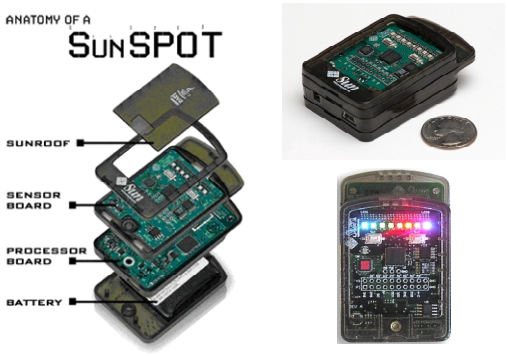

Figure 3.1 Sun SPOT sensor

The wide application range of Sun SPOT includes robotics, environmental monitoring, asset tracking, proactive health care and many others. In our design, the Sun SPOT performs as the acceleration sensor and the data processing center to recognize people's daily physical activity and detect fall.

The reasons we select the Sun SPOT because the hardware platform includes: (i) The tri-axis accelerometer with maximum -6G to +6G acceleration range is able to detect people's daily physical activity and unexpected fall; (ii) The on-board 180MHz, 32-bit ARM920T microprocessor is able to implement complex algorithm; (iii) Its light weight and small size make it easy to be attached to the testing subjects; (iv) The battery is able to keep the system running for a maximum of 7 hours without recharging; (v) The 8 multi-color LEDs on board can be used to display the detection result; no other display equipment is required.

The Sun SPOT device application development platform is supported via standard IDEs such as NetBeans. Also the Sun SPOT Software Development Kit (SDK) provides the necessary documentation, code samples and software to help users develop applications for SPOTs [23].

On the sensor board, a LIS3L02AQ accelerometer is embedded, which consists of a Micro-Electro-Mechanical System (MEMS) sensor element. When the accelerometer leans from its nominal position, the MEMS sensor causes an electrical imbalance that is read via the demo sensor board's analog-to-digital converter. The raw voltage value is then converted to g-force units. The accelerometer can be set to measure accelerations over a scale of either ± 2g or ± 6g [24]. In order to detect fall in a big range of acceleration, the acceleration is configured to a scale of ± 6g.

Figure 3.2 shows the components involved in reading acceleration from the accelerometer. The MEMS sensor of the LIS3L02AQ accelerometer is capable of measuring acceleration over a maximum bandwidth of 4.0 KHz for the X and Y axis and 2.5 KHz for the Z axis [24]. After the sample and hold process in the LIS3L02AQ accelerometer, the data go through a single-pole low-pass filter to reduce noise from each output pin of the accelerometer. This low-pass filter is a 0.01μF capacitor on Sun SPOT sensor board with a cutoff frequency of 160Hz. The ADT7411 analog-to-digital converter can perform a conversion at maximum sampling rate of 22.2 KHz. Finally the digital signals are further processed in the ARM9 processor.

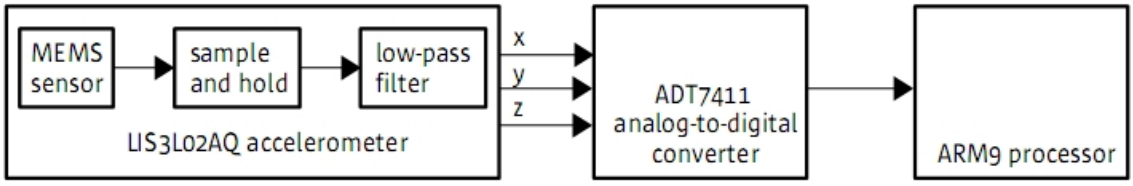

Figure 3.2 Components involved in reading the accelerometer [24]

There are 2 working modes depending on the various applications. Mode 1 is designed for indoor application and mode 2 is designed for outdoor application.

3.1.2 Mode 1: wireless sensors with a laptop computer

Mode 1 operation is shown in Figure 3.3. The two Sun SPOT sensors are attached to the chest and to the right thigh of a testing subject. Sensor 1 (i.e., the master sensor) receives raw data from sensor 2 (i.e., slave sensor), processes all the data to generate

classification results, and sends the results and raw acceleration data to a receiver. The receiver is connected to a laptop computer via USB interface. In the laptop terminal, once the receiver catches the coming data, data will be displayed on the monitor in real time. User can monitor the movement of a testing subject and record related data in designated files for further processing. Mode 1 is particularly designed for indoor application, whose user interface is designed on the laptop.

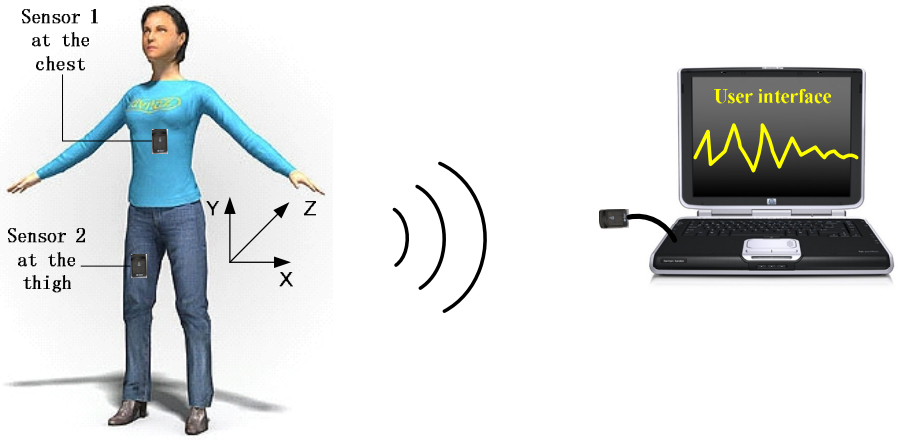

Figure 3.3 Mode 1: wireless sensors work with a laptop computer

3.1.3 Mode 2: wireless sensors without a laptop computer

Figure 3.4 shows the Mode 2 operation in which the two Sun SPOT sensors work without a laptop computer. The same as in Mode 1, the two sensors are still attached to the chest and to the right thigh respectively. Sensor 1 (master sensor) receives raw data from sensor 2 (slave sensor), processes all the data to generate a classification results. However, the results are not sent to other device. The eight multicolored LEDs on the sensor board indicate recognition result or generate fall alert signal. For example, LED_1 represents pattern_1, LED_2 represents pattern_2, LED_7 indicates fall alert signal, etc. Once a particular pattern is detected, its corresponding LED will light up immediately. Although mode 2 cannot provide detailed acceleration data, it is primarily designed for

outdoor application and provides users flexibility (e.g., when the user is running or walking).

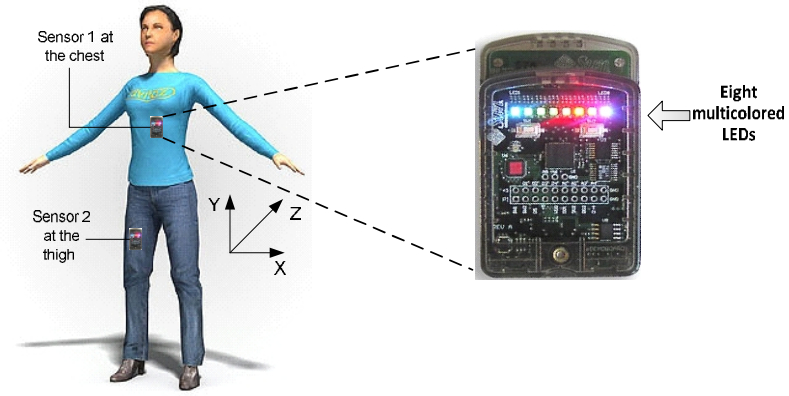

Figure 3.4 Mode 2: wireless sensors work without a laptop computer

3.2 Software component

In the sensor terminal, both training and testing processes are critical to make the system perform well. Figure 3.5 and Figure 3.6 show the block diagram of the training and testing progress respectively. In the training process, acceleration data of a known activity (e.g., simulated falls, sitting, standing, lying, walking, walking up the stairs) are firstly cut into small windows, and useful feature is extracted in each window. Then the data are used to build a Gaussian model for each attribute, and finally the Naive Bayes classifier is ready to test unknown data.

In the testing process, acceleration data of unknown activity are also cut into small windows with feature extraction. These data go through the Naive Bayes classifier to produce recognition result.

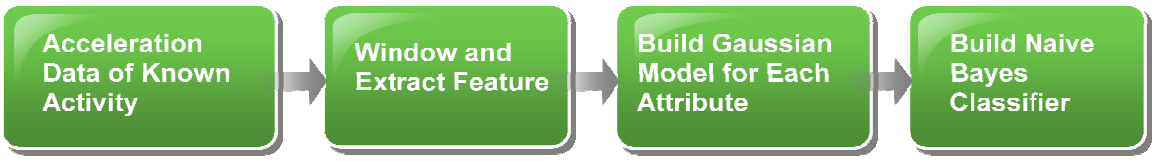

Figure 3.5 Block diagram of the training progress

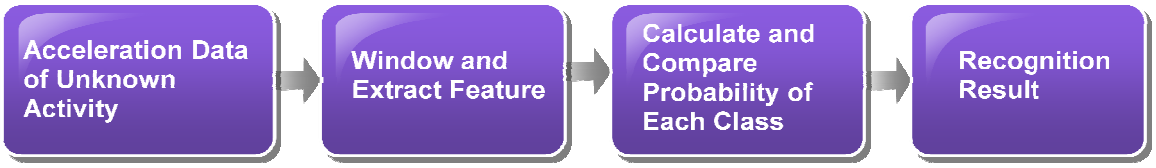

Figure 3.6 Block diagram of the testing progress

People's physical activities vary in a broad range, which depend on their age (e.g., the elderly walk slower than the young people), gender as well as different styles (e.g., the walking style varies among people); even the same person performs different physical activity at different time. Additionally, various falling speed and direction increase the hardship of fall detection. Based on the strategy of training and testing, a machine learning approach is supposed to make the system adapt to various users. During the training process, the characteristics of physical activity of a particular user are extracted and recorded by the system so that the testing results are more accurate and reliable. The laptop computer terminal receives raw acceleration data, classification result and displays the results on a GUI.

Chapter 4 Methodology

In this chapter, we describe how the raw acceleration data collected from the accelerometer are processed and physical activity is recognized and fall is detected. First, the raw data are pre-processed in the time domain to window into small segments with a fixed window size. Then the specific feature is extracted from each window for further processing. Finally, an optimal classifier is selected from popular machine learning algorithms; WEKA (Waikato Environment for Knowledge Analysis) [25] is used as a comparison platform, in which off-line simulations are performed based on the acceleration data collected from the sensors.

4.1 Windowing technique

Before raw acceleration data going through the Naive Bayes classifier, raw acceleration data must be pre-processed using a windowing technique in order to increase classification accuracy. Windowing technique is used to divide the sensor signal into smaller time segments (i.e., windows) and classification algorithm is applied separately on each window, which means each window produces a classification result. Without the window technique, algorithms have to process each sample data, which becomes ineffective (e.g., the FFT algorithm is able to extract useful information in frequency domain within an appropriate length of window size rather than a single data sample).

4.1.1 Three popular windowing techniques

Three different windowing techniques have been used in activity monitoring, which are sliding windows, event-defined windows and activity-defined windows [2].

1) **Sliding window:** With the sliding window method, the signal is divided into windows of fixed length with no inter-window gaps, as shown in Figure 4.1. The window size (i.e., the number of samples) is dependent on particular application. For gently changing signals (e.g., sitting still on the chair), a bigger window size is adequate; for intensely changing signals, such as an unexpected fall or running on the athletic ground, a smaller window size obviously works better. The sliding window approach does not require complex processing of the sensor signal and is therefore ideally suited to real-time applications. Due to its implementation simplicity, most activity classification studies have employed this windowing technique [2].

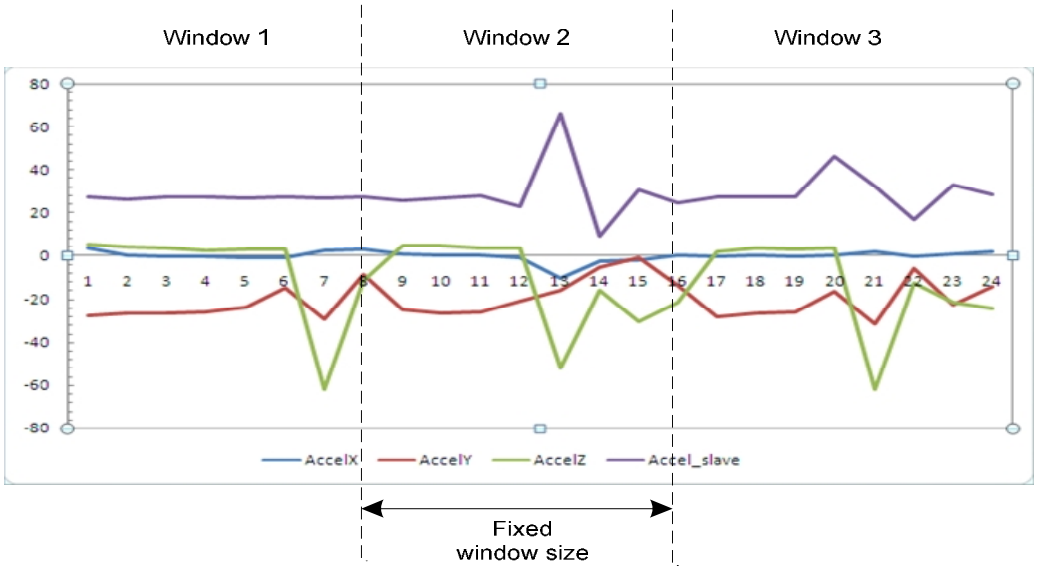

Figure 4.1 Sliding windowing technique: fixed window size

2) **Event-defined window:** In event-defined windows, pre-processing is required to locate specific events, such as heel strike or toe-off. Once the events are located successfully, windows are defined successively. Provided that the duration of these events vary in time domain, the size of these windows is not fixed. In order to identify heel strike and toe-off, different approaches have been proposed [2].

3) **Activity-defined window:** Activity-defined window is another approach, where the most critical technique is to identify the activity transition points, which are then used to define windows of raw data. For example, the acceleration data will change intensively when the user starts to run from a stand still position. Once the activity transition point is determined by a certain algorithm, the two windows representing running and standing are defined respectively. A number of methods have been proposed to identify activity-transition points prior to explicitly identifying the specific activities [2].

4.1.2 Selecting sliding window approach

The three approaches mentioned above have their own applications. As such, in order to recognize physical activity and detect fall in real time, the sliding window approach with a fixed window size is chosen since it is simple to implement into embedded systems. The window size for pre-processing is very important in detection accuracy. Small window will misclassify pattern while big window has a potential to misclassify a fall (e.g., if falling time is much less than the window time, the fall will be ignored). Large window size has low performance in fast movement recognition.

After numerous experiments, it has been found that 0.2 second is a good window time for detecting fall, which means a classification result is produced in every 0.2 second. At a 20Hz sampling rate, the window size is 4 samples. As mentioned before, the characteristics of fall is different among different individuals, so a specific window time maybe meets the requirement of other individuals.

4.2 Feature extraction

In each small time domain segment, useful features are to be extracted as the input to the classifier. Fall and physical activity have some specific features which can be used to distinguish one from others.

4.2.1 Features of fall and physical activity

Due to the characteristics of falls, such as maximum or minimum peak in time domain, falls can be differentiated from normal physical activity, such as standing or walking.

Figure 4.2 shows the fall orientations: forward, backward, leftward and rightward and how the sensors are attached on the testing subject.

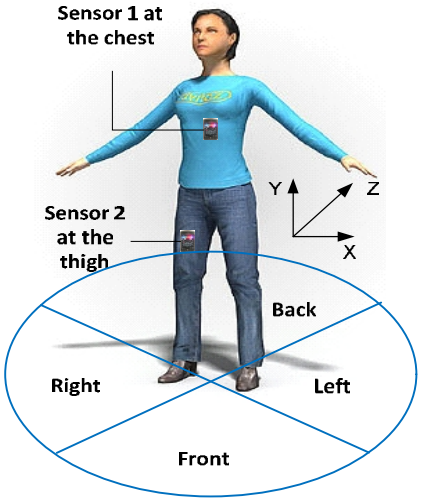

Figure 4.2 Fall orientations

Figure 4.3 illustrates the characteristics of falls, in which each type of fall is trained 3 times respectively. All the data in Figure 4.3 are collected by Sensor 1 at the chest. The horizontal axis is sample window, and the vertical axis is the acceleration value before normalization. The peaks in Figure 4.3 displays forward, backward, leftward and rightward fall very well. Forward (Peak_1 to Peak_3) and backward fall (Peak_4 to Peak_6) have obvious peaks in acceleration Z axis (dash line in green color), negative and positive peak respectively. Leftward (Peak_7 to Peak_9) and rightward fall (Peak_10 to Peak_12) have obvious peaks in acceleration X axis (solid line in blue color), positive and negative peak respectively. In addition, acceleration Y axis (dot line in red color) cannot be used to classify different falls, because each type of fall has a positive peak in Y axis direction. However, this attribute is able to detect falls from normal physical

activity efficiently. During the training of falls, all the peak values are recorded to build models in classifier.

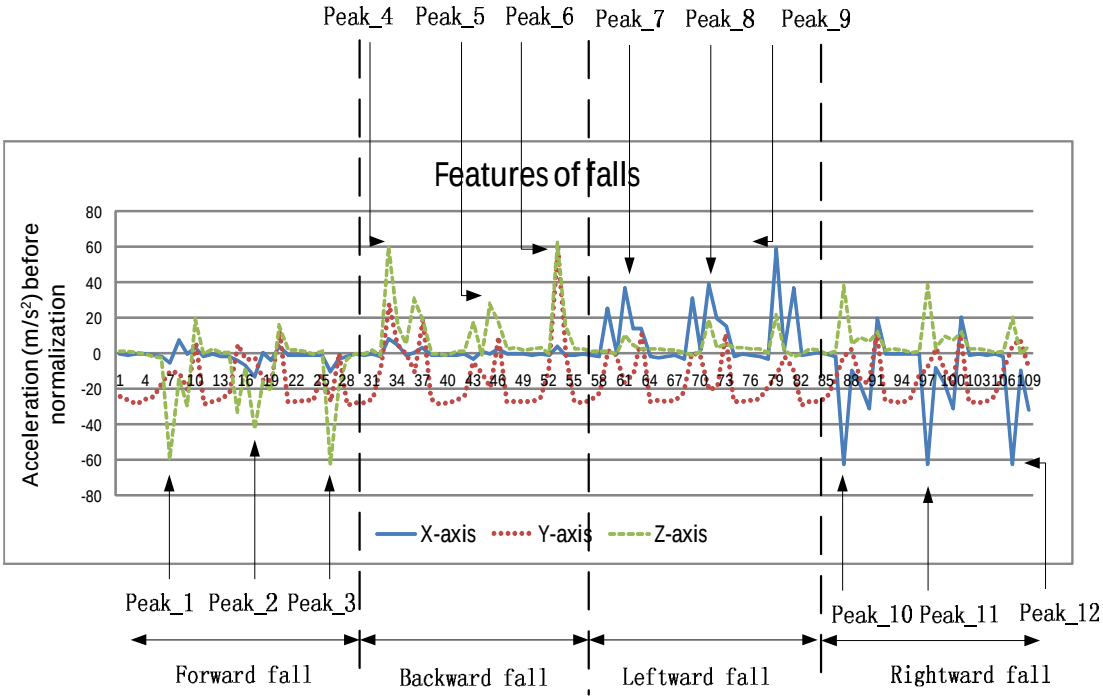

Figure 4.3 Features of falls. Each peak represents a fall

In terms of physical activities, Figure 4.4 shows the average acceleration data before normalization. The acceleration data are collected from one sensor attached to the thigh of a testing subject. Each pattern is sampled 100 data, and then calculated the mean value. Based on various acceleration values in three axes, five physical activities can be differentiated. Obviously, the static postures are easy to be classified, but it is possible to misclassify walking and walking up the stairs.

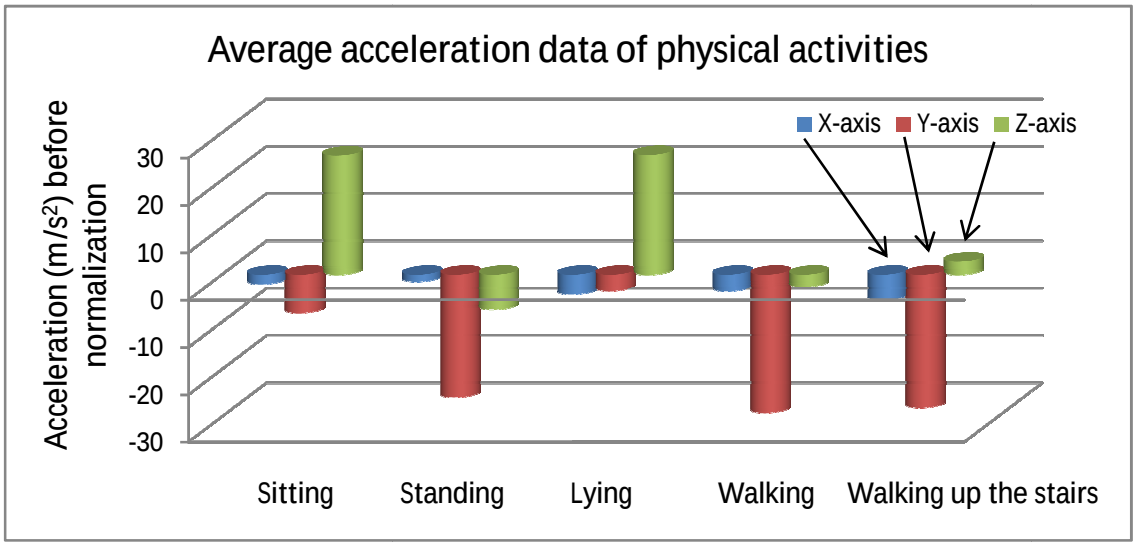

Figure 4.4 Average acceleration data of each physical activity

4.2.2 Feature extraction approaches

Numerous approaches have been proposed to generate features which characterize windows of acceleration data. The features rather than raw data are processed by specific classification algorithm. Before advanced classification algorithms are applied to features, some methods for selecting optimal features from a larger set and methods for reducing dimensionality of features can be used to process the data [2].

Time-domain features: Time-domain features are derived directly from a window of acceleration data. Popular features include static values such as mean, root mean square, standard deviation, etc.

Frequency-domain features: The window of raw data must be transformed into frequency domain to produce the frequency-domain features. Fast Fourier transform (FFT) is an effective algorithm to accomplish this transition. The output of a FFT typically generates a group of coefficients which represent the amplitudes of the frequency components of the signal as well as the distribution of the signal energy [2]. For periodical movement, such as running and walking, frequency-domain features indicate more useful information than time-domain features.

Wavelet analysis (time-frequency features): The advantage of wavelet analysis is to investigate both time and frequency characteristics of a signal. Discrete wavelet transform (DWT) is mostly employed in activity monitoring. The wavelet analysis has been applied to three different types of issue in activity monitoring, which are signal enhancement, identification of transition points and generation of time-frequency features [2].

In this real-time application, time-domain feature performs better than frequency-domain feature or wavelet analysis due to the limitation of computation resource. Some time-domain features (mean, root mean square, and standard deviation) are compared and evaluated. Table 4-1 shows the time-domain feature definition. After simulation, the mean works better than others, with a higher accuracy.

Table 4-1 Time-domain feature definition

Feature	Definition
Mean	$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$
Root mean square	$x_{rms} = \sqrt{\frac{x_1^2 + x_2^2 + \ldots + x_n^2}{n}}$
Standard deviation	$s = \sqrt{\frac{(x_1 - m)^2 + \ldots + (x_n - m)^2}{n}}$ where m is the mean value

As shown in Figure 4.5, every 4 samples (i.e., S1 to S4) make up one window. On each window, the mean value is calculated to produce mean_1, mean_2, etc. As time rolls on, the window jumps forward 0.2 second (i.e., 4 samples). Since this approach is very easy to implement, it is ideally appropriate for real-time application in embedded system.

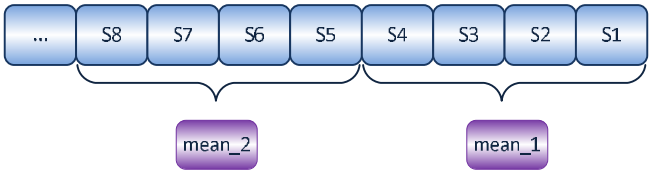

Figure 4.5 Four samples make up one window

4.3 Algorithm evaluation criteria

Evaluating the performance of different algorithms is critical. Based on the training data collected from previous steps, a classifier is built up to classify an unseen instance. In order to select the optimal algorithm, algorithm evaluation criteria are proposed. The evaluation is important for understanding the quality of the classifier and for choosing the most acceptable classifier [26].

Different criteria are used to evaluate the classifiers. Obviously, classifiers providing high accurate results are favorable. In terms of other performances, the computational complexity and the comprehensibility are two important aspects.

1) **Accuracy:** Accuracy is the degree of closeness of measurements of a quantity to its actual (true) value. In our application, accuracy is equal to correctly classified measurements divided by total measurements. High accuracy provides more reliable and satisfied classification results. In the application of fall detection, whether the system can successfully detect falls depends on the accuracy of the classifier. Low accuracy cannot guarantee the protection system will perform correctly when a fall really happens.
2) **Computational complexity:** Computational complexity is another important criterion for comparing algorithms. It is defined as the amount of CPU consumed by each classifier [26]. The computational complexity is described in three aspects.
    a) Computational complexity for generating a new classifier: When the training data have many attributes and massive instances, the complexity is prohibitively expensive to generate the classifier based on the provided training data. For example, Neural Network algorithm has more computational complexity than OneR (one level decision tree) in generating a new classifier.
    b) Computational complexity for updating a classifier: It is necessary to update the current classifier with new data to make it properly adapt to users.
    c) Computational Complexity for classifying a new instance: In real time application, this type of complexity is the most critical factor. Computationally expensive algorithm limits classification speed, and even misses useful data when the microprocessor is calculating.

3) **Comprehensibility:** Comprehensibility criterion refers to how well humans grasp the classifier. In some applications, users not only require high accuracy, but also intend to interpret how the classifier makes decisions, which benefits non professionals to be confident of the classification results. Some classifiers with complex algorithms are considered as black-box since they are difficult to understand, such as neural networks and support vectors machines [26]. But classifiers like one level decision tree are easy to interpret, referred to transparent-box algorithm.

As described above, the accuracy and complexity can be quantitatively estimated, but the comprehensibility is more subjective.

4.4 Popular data mining algorithms

The theories of some popular artificial intelligence algorithms are described, including decision tree, Support Vector Machines (SVM), Artificial Neural Network (ANN) and Naive Bayes. The advantages and disadvantages of each method are discussed.

4.4.1 Decision tree: C4.5

Decision tree is a popular type of data mining algorithm. It consists of nodes and branches connecting the nodes. The top node of the tree, defined as the root, includes all the training data, which are finally split to classes. The bottom nodes of the tree are called the leaves, indicating classes. All nodes except the leaves are defined as decision nodes, where training examples are split into distinct classes based on one attribute [27]. In the testing progress, every new testing data goes into a specific branch from the root, following a matching path to a particular leaf.

One example is given to demonstrate how this algorithm works. For students live in Saskatoon, we assume that picking Saskatoon berry is an event which depends on three attributes (features): weather (F1), availability (F2), and berry farm (F3). Rainy day is not

suitable for outdoor activity; heavy work (preparing for exam or seminar) is also not suitable; if the berry farm is closed, picking berry is unavailable. Table 4-2 shows the training data for building up a decision tree. Eight set of data are collected, from D1 to D8.

Table 4-2 Training data for building up a decision tree

	F1	**F2**	**F3**	**F4**
Data	Weather	Availability	Berry farm	Pick Saskatoon berry
D1	Sunny(1)	Light work(1)	Open(1)	Yes
D2	Sunny(1)	Heavy work(2)	Open(1)	No
D3	Rainy(2)	Light work(1)	Open(1)	No
D4	Rainy(2)	Heavy work(2)	Open(1)	No
D5	Sunny(1)	Light work(1)	Closed(2)	No
D6	Sunny(1)	Heavy work(2)	Closed(2)	No
D7	Rainy(2)	Light work(1)	Closed(2)	No
D8	Rainy(2)	Heavy work(2)	Closed(2)	No

Based on the training data in Table 4-2, a decision tree is built up in Figure 4.6. At each decision node, a specific feature is used to split the data into different branches until the data reaches a leaf.

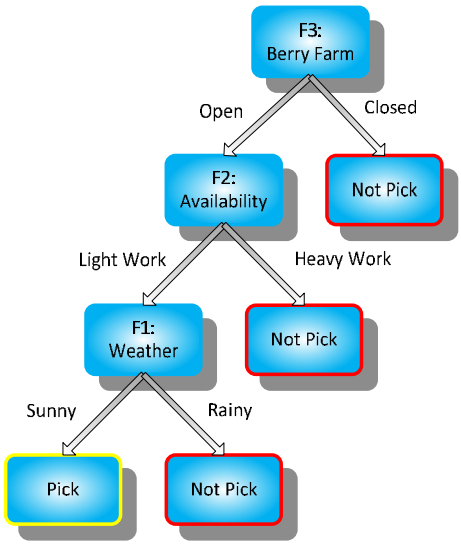

Figure 4.6 Decision tree for picking Saskatoon berry

There are numerous algorithms based on the decision tree principle. C4.5 is a typical algorithm, one of best-known and most widely-used learning algorithms. It is an extension of Quinlan's earlier ID3 algorithm, which uses Shannon's entropy as a criterion for selecting the most significant features:

$$Entropy\ (S) = \sum_{i=1}^{c} -p_i \cdot log_2(p_i) \tag{4.1}$$

where p_i is the proportion of the examples belongs to the i^{th} class [27].

At each decision node, available attributes are evaluated by information gain, which is equal to information entropy before splitting subtracts information entropy after splitting [27]. The attribute with greatest information gain is selected to split the data most effectively. The procedures are repeated until the data cannot be split any further. When constructing the decision tree, the time complexity for sorting is expressed as $O(n\log n)$. It can be estimated that the complexity drastically increases when there are many attributes and each attribute has many numeric values.

C4.5 includes the features: permit numeric attributes, deal sensibly with missing values, and prune to deal with noisy data. The last research version is C4.8, which is implemented in WEKA as J48 (Java).

The main advantages of the decision tree as a classification tool include [27]:

- Decision tree is self-explanatory and comprehensible, and even non-professional users can grasp it.
- Decision tree can handle both nominal and numeric input attributes.
- Some decision trees can handle both continuous and discrete attributes.
- Decision tree can handle datasets that may have errors, or missing values.

The main disadvantages include [27]:

- As decision trees use the "divide and conquer" method, they tend to perform well if a few highly relevant attributes exist, but less so if many complex interactions are present.
- Decision tree is limited due to its over-sensitivity to the training set, to irrelevant attributes and to noise.

4.4.2 Support Vector Machine (SVM)

Support vector machines (SVM) are supervised learning methods for classification and regression. Similar to other classifiers, the SVM classifier is firstly trained, and then unknown samples go through the classifier to be categorized. In the training progress, a hyperplane is constructed in a high or infinite dimensional space to classify data into one of the two categories (i.e., category 0 or 1). In the testing progress, it predicts whether a new sample falls into one category or the other.

Although more than one type of hyperplane can separate the data, an optimal hyperplane which has the largest distance to the nearest training data points of any class is selected. The geometric distance between two categories of training data is defined as the functional margin. Sine in general the larger the margin the lower the generalization error of the classifier, a good algorithm should maximize the functional margin under some constraints.

For simple linear SVM, the optimal hyperplane is achieved by solving the following optimization problem. Find a vector w to maximize the margin, the same as minimize $\Phi(w) = \frac{1}{2}w^t w$, subject to $y_i(wx_i + b) \geq 1 \ \forall i$. The details of SVM algorithm can be found in a tutorial [28]. In some applications, non-linear SVM works better because the original input space can always be mapped to some higher-dimensional feature space where the training set is separable. In the transformation $\Phi: \ x \rightarrow \phi(x)$, a kernel function is used to correspond to an inner product in some expanded feature space. Typical kernel functions include:

- Linear: $K(x_i, x_j) = x_i^T x_j$
- Polynomial of power p: $K(x_i, x_j) = (1 + x_i^T x_j)^p$
- Gaussian (radial-basis function network): $K(x_i, x_j) = \exp(-\frac{\| x_i - x_j \|^2}{2s^2})$
- Sigmoid: $K(x_i, x_j) = \tanh(b_0 x_i^T x_j + b_1)$

The best choice of kernel function for a given problem is still a research issue.

Unlike multi-class applications such as neural networks or even multiclass support vector machines (MSVMs), simple SVM can only classify data with a binary strategy. However researchers expand simple SVM to make it handle multi-class problems. One

approach is to train and test data in the one-against-all method, which separates one class from the others. Provided a k classes problem, one must build k SVM models. Another common approach is only to distinguish between two classes (the one-against-one method). Provided a k classes problem, we must build $\frac{k\cdot(k-1)}{2}$ SVM models.

SVM has been used successfully in many real-world problems, such as text (and hypertext) categorization, image classification, bioinformatics (protein classification, cancer classification) and hand-written character recognition.

The advantages of SVM include [29]:

- By selecting the kernel function, SVM is flexible of choosing the form of the threshold distinguishing one class from another.
- It does not need to represent the space explicitly, simply by defining a kernel function.
- SVM produces a unique solution, since the optimality problem is convex.

The weakness of SVM includes [28]:

- It is sensitive to noise. A relatively small number of mislabeled examples can dramatically decrease the performance.
- The best choice of the kernel function for a given problem is still a research issue. Even a certain kernel function has been fixed, the relevant parameters are not easy to decide, such as the Gaussian kernels the width parameter, as well as the value of ε in the ε-insensitive loss function.
- The speed and size of SVM both in training and testing limit SVM's performance. The high algorithmic complexity and extensive memory requirements of the required quadratic programming in large-scale tasks cannot satisfy real-time applications.
- The optimal design for multiclass SVM classifiers needs further research.

4.4.3 Neural Network

Neural network is another popular algorithm that can be used for data classification. Inspired by the working principle of human brain, which basically learns from experience

based on myriads of neurons, researchers propose artificial neural network. It is a mathematical model to simulate the structure and functional aspects of biological neural networks. An interconnected group of artificial neurons (called nodes) as well as the weight values make up of the artificial neural network structure. Figure 4.7 is a graph of a simple neural network with three layers (input layer, hidden layer and output layer). Changing of its connection weights (training) causes the network to learn the solution to a problem. The strength of connection between the neurons is stored as a weight value for the specific connection. The system learns new knowledge by adjusting these connection weights. The learning ability of a neural network is determined by its architecture and by the algorithmic method chosen for training [30]. Neural network is an adaptive non-linear statistical data modeling system.

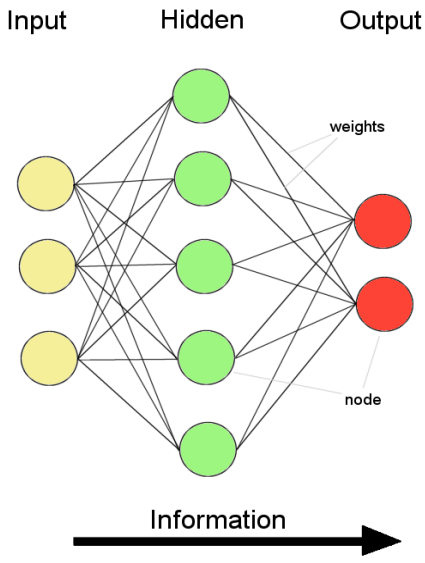

Figure 4.7 A simple neural network with three layers [31]

In the training progress, data go through hidden layer to determine the weight value between neurons. The weight values are assigned random values at the beginning, and during the training progress it can be adjusted by the difference between target output and actual output. $\Delta w_i = \eta(t - o)x_i$, where t is the target output, o is the actual output generated by the perceptron and η is a positive constant as the learning rate. The Δw_i is used to update the weight value (i.e., $\Delta w_i + w_i \rightarrow w_i$). After training, each weight value

between neurons is fixed. In the testing progress, with the fixed weight value, unknown data enter the neural network to calculate the actual output, which represents the class it belongs to.

Since the neural network can be used to recognize and match complicated, vague, or incomplete patterns, it has a wide application range. Prediction (learning from past experience, e.g., predict weather), classification (e.g., image processing), recognition (handwriting recognition), data association, data conceptualization, data filtering as well as planning are concrete applications.

The advantages of neural network include:

- It can adapt to unknown situations.
- It is robust with fault tolerance due to network redundancy.
- It is capable of autonomous learning and generalization.

The disadvantages include:

- Low comprehensibility because the whole processing is considered to be a black box, not easily to interpret.
- When the input data increase, the computational complexity of the structure of the network increases dramatically. For real-time classification, large computational complexity limits the processing speed both in training and testing.

4.4.4 Naive Bayes

Naive Bayes classifier is a simple probabilistic classifier based on Bayes' rule. The probability of event H given evidence E is represented as $p(H|E) = \frac{p(E|H)\cdot p(H)}{p(E)}$, where *p(H)* is the probability of event before evidence is seen, and *p(H|E)* is the probability of event after evidence is seen. Provided *p(E|H)*, *p(H)* and *p(E)* are known, *p(H|E)* can be easily calculated.

It is assumed that all attributes of an instance are equally important and statistically independent. Although the independence assumption is incorrect (e.g., the tri-axis acceleration values in the master sensor have specific relationship), this Bayes model works well in practice.

In the physical activity recognition application, four attributes and five classes are important parameters, as shown in Table 4-3. Five classes include sitting, standing, lying, walking and walking up the stairs, and the four attributes are A1 to A4.

Table 4-3 Parameters of the Naive Bayes classifier

	Number	**Specification**
Class (C)	5	C1: sitting, C2: standing, C3: lying, C4: walking, C5: walking up the stairs.
Attribute (A)	4	A_1, A_2, A_3:acceleration x, y, z from the master sensor, A_4: $\|a\|=\sqrt{a_x^2+a_y^2+a_z^2}$ from the slave sensor.

The actual Naive Bayes classifier is shown in the following probability model [32]:

$$p(C \mid A_1, \mathbf{K}, A_n) = \frac{1}{Z} p(C) \prod_{i=1}^{n} p(A_i \mid C) \tag{4.2}$$

in which Z is a scaling factor depending only on A_1 through A_n, C is the class variable, A_1 through A_n are independent attributes variables, and *p(C)* is called class prior probability.

Since each instance has four attributes which are all numeric acceleration data (e.g., acceleration value in x-axis is 0.65 g), we assume that all the attributes have a normal or Gaussian probability distribution. The probability density function for the normal distribution is defined by two parameters:

Sample mean:

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{4.3}$$

Standard deviation:

$$\sigma = \sqrt{\frac{(x_1-\mu)^2+\cdots+(x_n-\mu)^2}{n}} \tag{4.4}$$

Then the density function f(x) is:

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{4.5}$$

The mean and standard deviation depend on the attribute of Class C. Then after the training process, all the related means and standard deviations are calculated and the

Gaussian probability distribution models are built up. Each class has four attributes and there are five classes, thus we build up 20 Gaussian models. In the testing process, given a particular raw data, five probabilities are calculated and compared, i.e., $p(sit|A_1, A_2, A_3, A_4)$, $p(stand|A_1, A_2, A_3, A_4)$, $p(lie|A_1, A_2, A_3, A_4)$, $p(walk|A_1, A_2, A_3, A_4)$ and $p(walk\_up|A_1, A_2, A_3, A_4)$.

One example for calculating sitting probability is shown below:

$$\mathrm{p(sit|A_1, A_2, A_3, A_4)} = \mathrm{p(sit)} \cdot \mathrm{p(A_1|sit)} \cdot \mathrm{p(A_2|sit)} \cdot \mathrm{p(A_3|sit)} \cdot \mathrm{p(A_4|sit)} \quad (4.6)$$

$$\mathrm{p(stand|A1, A2, A3, A4)} = \mathrm{p(stand)} \cdot \mathrm{p(A_1|stand)} \cdot \mathrm{p(A_2|stand)} \cdot \mathrm{p(A_3|stand)} \cdot \mathrm{p(A_4|stand)} \quad (4.7)$$

……

where $p(sit)$=1/5 and $p(A_1|sit)$ to $p(A_4|sit)$ have a Gaussian distribution. If $p(sit|A_1, A_2, A_3, A_4)$ has the highest probability than others, a decision is made that the particular raw data belongs to class 1, which means the subject is sitting.

Naive Bayes classifier works surprisingly well (high accuracy), even if the attributes are not completely independent. The classification doesn't require accurate probability estimates as long as maximum probability is assigned to correct class. The computational complexity is also low, suitable for real-time application.

However, too many redundant attributes will increase the computational complexity and decrease the classification accuracy. The most important attributes should be selected. In addition, many numeric attributes are not normally distributed, so we have to use kernel density estimators depending on the actual application.

4.5 Algorithm evaluation in WEKA

The 4 classifiers discussed above have advantages and disadvantages. In order to select an optimal algorithm with high recognition accuracy as well as low complexity in system implementation, the algorithms are compared and evaluated. In real-time

application, time is a crucial parameter to evaluate algorithms which recognize physical activity and detect fall. The selected algorithm should not only detect falls as accurate as possible, but also generate fall alert signal as fast as possible. Low accuracy of fall detection is not able to prevent falls; slow algorithm needs more time for calculation so that protection system becomes ineffective.

4.5.1 Introduction to WEKA

The tool we use for comparing different algorithms is WEKA (Waikato Environment for Knowledge Analysis), developed at the University of Waikato, New Zealand. WEKA provides an extensive collection of machine learning algorithms and data preprocessing tools, which enables the comparison of different machine learning methods on new data sets. The algorithms in WEKA can be used for regression, classification, clustering, association rule mining and attribute selection. The data to be processed are described by a fixed number of attributes (numeric or nominal attributes and some other types) [25].

One of WEKA's strengths is that it contains a collection of visualization tools and algorithms for data analysis and predictive modeling. Machine learning algorithms such as Artificial Neural Network, Decision Tree and Naive Bayes can be easily realized in WEKA. All classifiers in WEKA are fully implemented in Java, which enables the tool run on almost any modern computation platform. WEKA's source code is easy to access due to its open source policy.

Figure 4.8 is the WEKA Explorer user interface. Four attributes are AccelX_thigh, AccelY_thigh, AccelZ_thigh and Accel_chest, where the first three attributes are acceleration data from the sensor attached to the thigh, and Accel_chest is the acceleration data from the sensor attached to the chest. Five classes include sit, stand, lie, walk and walk up the stairs. Forty training data of each class are imported to build up the classifier model, represented by class histogram in Figure 4.8. The graphic user interface enables the data to be processed by different classifiers.

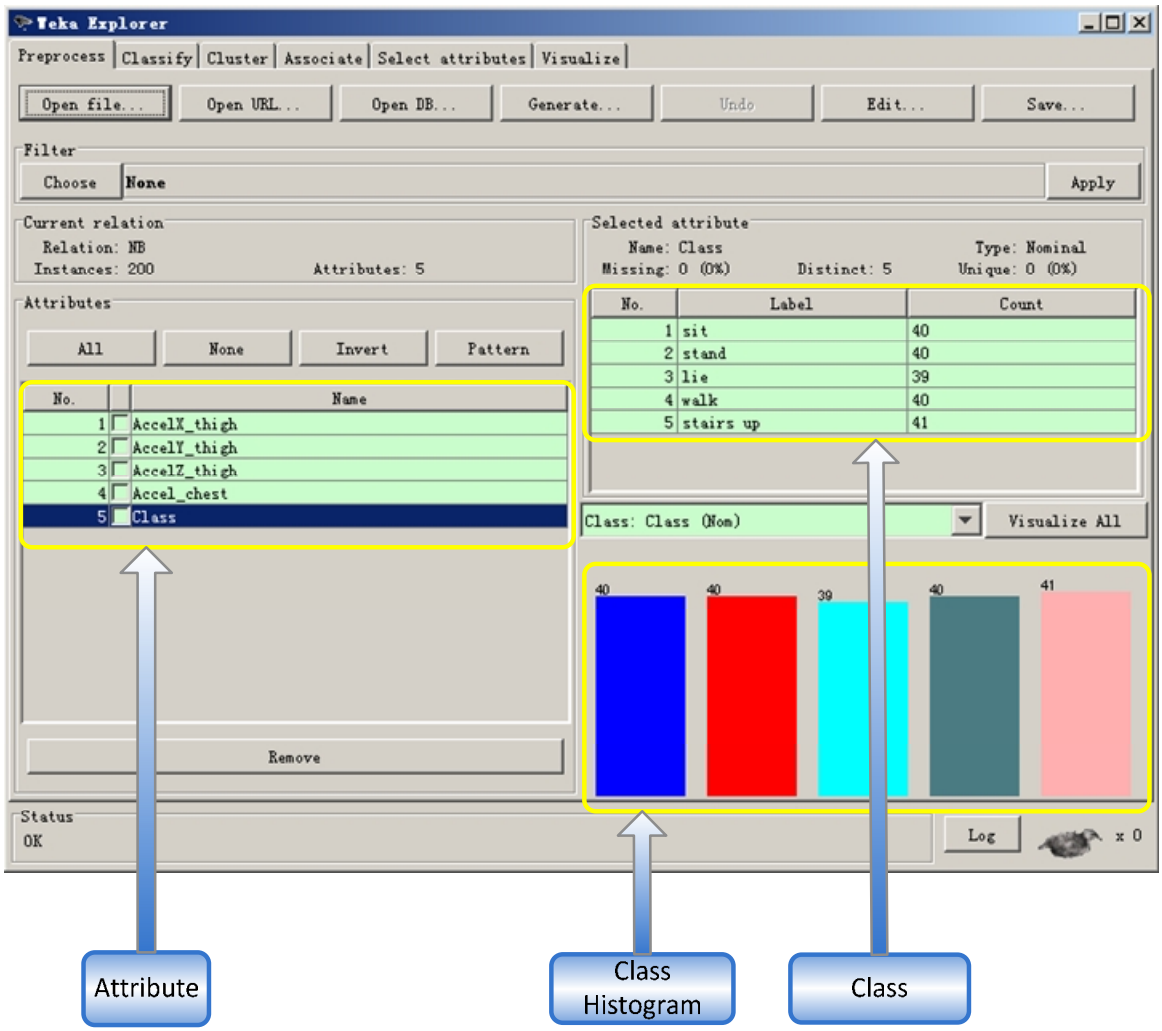

Figure 4.8 The WEKA Explorer user interface

4.5.2 Algorithm comparison result

In order to evaluate the algorithms mentioned above, an experiment environment is set up. Two Sun SPOT sensors with tri-axis accelerometers are attached to the chest and the right thigh of a 23 years old female subject. The acceleration data collected from those sensors are processed offline in WEKA, and all algorithms perform with the default setting. The laptop running WEKA is a 1.6GHz processor with 512 MB memory.

In physical activity recognition, compared with threshold-based method, artificial intelligence method has more advantages both in accuracy and implementation. Obviously calibration is needed in threshold-based method, and fixed threshold cannot adapt to the variance of different people, which decreases the system accuracy. Some popular artificial intelligence methods, including ZeroR, Support Vector Machine (SVM), OneR, C4.5, Neural Network and Naive Bayes, are compared and evaluated in WEKA. Six algorithms run with exactly the same input data: 4 attributes, 200 train data and 486 test data. There are 5 patterns to be trained and tested: sitting, standing, lying, walking and walking up the stairs.

Table 4-4 shows the algorithms comparison results. In terms of accuracy, Naive Bayes algorithm overtakes the others with 87.9% accuracy, while Support Vector Machine (SVM) is 64.40%, and ZeroR is only 20.6%. As an embedded system, less power consumption is required. The power consumption increases proportionally as long as a complicated algorithm is used as the real-time classifier, because it always consumes more CPU cycles in each classification progress. Also the parameter of "time to build the model and generate classification results" is critical for evaluating algorithm's implementation. SVM algorithm consumes 2190ms to build the model, while Naive Bayes algorithm consumes very little time. Therefore, Naive Bayes algorithm is selected as the optimal algorithm with a relatively high accuracy as well as less processing time.

Table 4-4 Algorithms comparison results

Algorithm	Algorithm Name in WEKA	Correctly Classified Instances (%)	Incorrectly Classified Instances (out of 486)	Time to build model and generate classification results (millisecond)
ZeroR	rules.ZeroR	20.6	386	<1
SVM	functions.SMO	64.4	173	2190
OneR	rules.OneR	74.7	123	<1
C4.5	J48	81.1	92	550
Neural Network	functions.Multi-layerPerceptron	84.6	75	940
Naive Bayes	bayes.NaiveBaye Simple	87.9	59	<1

Chapter 5 Implementation

In this chapter, implementations of the system in the Sun SPOT sensors and in the laptop computer are described. The program flow chart in the master sensor, the training and testing strategies are also provided. In the laptop terminal, a graphic user interface (GUI) is used to display the acceleration data in real time. Figure 5.1 is the block diagram of the system set-up for Mode 1 and Mode 2. Mode 2 is in the dash box, in which Sun SPOT Slave sensor sends acceleration data to the Master sensor wirelessly, and the Master sensor processes data to perform classification. Mode 1 is illustrated in the solid box. In Mode 1, Sun SPOT Slave sensor sends acceleration data to the Master sensor; Master sensor accomplishes all the calculation and sends the classification results as well as raw acceleration data to the Sun SPOT receiver; then the Sun SPOT receiver is connected to the laptop computer via a USB interface. Classification results and raw acceleration data are displayed in the laptop computer.

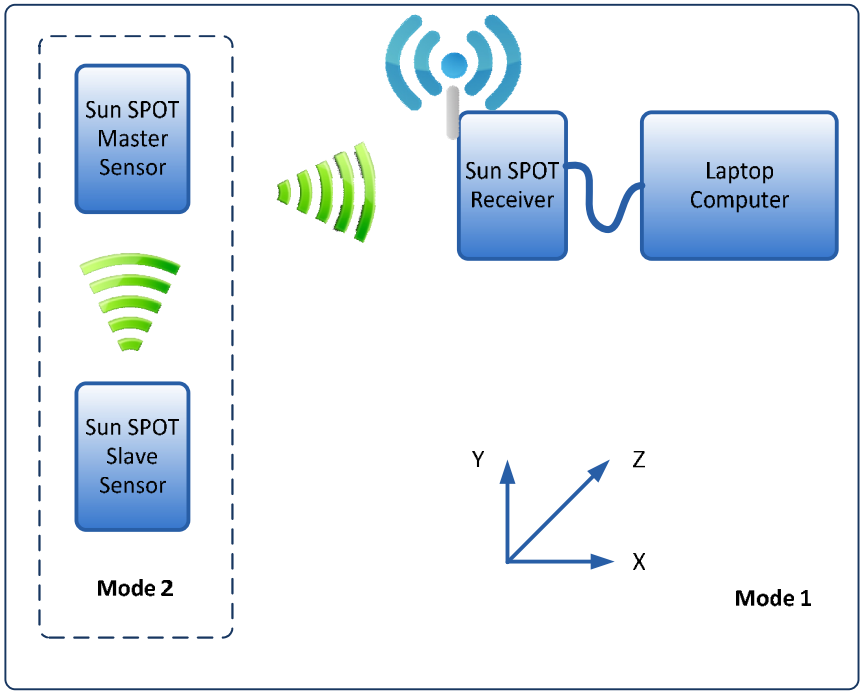

Figure 5.1 Block diagram of the system set-up for Mode 1 and Mode 2

5.1 Implementation on Sun SPOT

Sun SPOT sensors are the core processing units. Slave sensor and master sensor work together to collect and process raw acceleration data, and finally produce classification results. As mentioned in Chapter 4, there are 4 attributes required in this application, where the master sensor provides 3 attributes (X-axis, Y-axis and Z-axis accelerations), and the slave sensor provides 1 attribute ($|a| = \sqrt{a^2 + b^2 + c^2}$).

5.1.1 Slave sensor

The only task the slave sensor performs is to sample the raw acceleration data at a specific rate then send to the master sensor. No calculation or further processing is performed. Once the switch 1 (i.e., pushbutton 1) on the slave sensor board is activated,

the slave sensor starts sampling data and sends. A red LED lights up to indicate working status of slave sensor. Figure 5.2 is the pseudocode in the slave sensor.

```
Monitor switch 1 (i.e., pushbutton 1) on slave sensor
    If switch 1 is pressed
        LEDs[7] lights up in red
        Get roundDouble data accel.getAccel()        //Get the forth attribute data
Send data to master sensor                           //Send data to master sensor
Sleep (1000/SAMPLE_RATE);                            //SAMPLE_RATE=20Hz
    Endif
```

Figure 5.2 Pseudocode in the slave sensor

5.1.2 Master sensor

The master sensor is the core processing unit, where all the calculation and classification are accomplished. In both training and testing process, the master sensor firstly receives the fourth attribute from the slave sensor, and then implements further processing and calculation on the collected data. Figure 5.3 shows the flow chart of the master sensor.

1) Training:

Switch 1 is used to start the training process, and five patterns (sitting, standing, lying, walking and walking up the stairs) are trained serially. After that, switch 2 is used to start the testing process. Pseudocode in Figure 5.4 is for monitoring the switch press. The LEDs[0] indicates pattern 1. Once training of pattern 1 finishes, the color of LEDs[0] will change from red to blue.

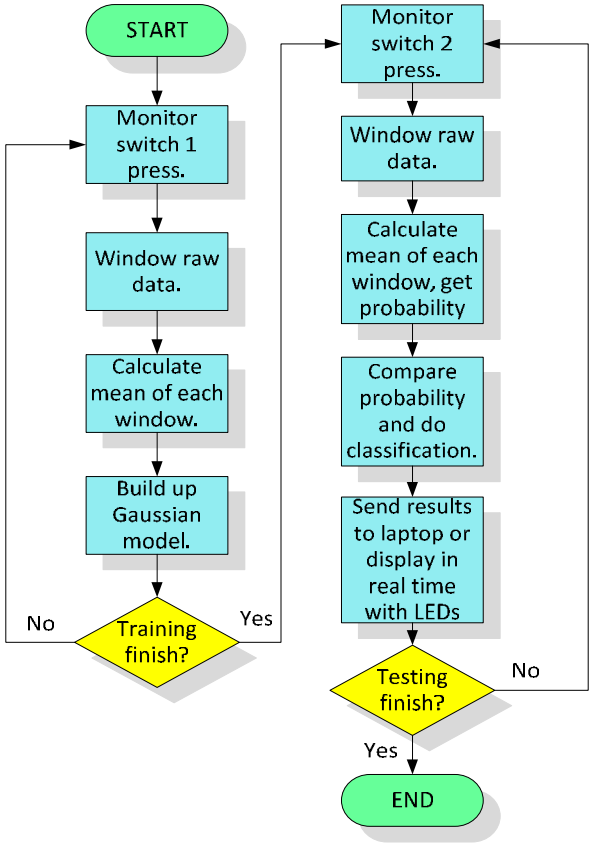

Figure 5.3 Flow chart of the master sensor

```
Monitor switchPressed (ISwitch sw)          //Monitor switch 1 and switch 2
      If switch 1 is pressed                // switch 1 starts training, train patterns serially
         LEDs[0] lights up in red           //LEDs[0] color is red when training
Train pattern 1
LEDs[0] lights up in blue                   //LEDs[0] color becomes blue after training
LEDs[1] lights up in red
Train pattern 2
LEDs[1] lights up in blue
…
LEDs[4] lights up in red
Train pattern 5
LEDs[4] lights up in blue
Else if switch 2 is pressed                 // switch 2 starts testing
   LEDs[7] lights up in red                 //LEDs[7] color is red when testing
   Test data and produce results            // Call test function
         LEDs[7] lights up in blue          //LEDs[7] color becomes blue after testing
      Endif
```

Figure 5.4 Pseudocode to monitor the switch press for both training and testing progress

In each training process, the raw acceleration data are cut into small windows, and the mean of each window is calculated. All these mean values are used to build a Gaussian model, where mean and standard deviation are two critical parameters. Figure 5.5 is the code to train data. Variable "m1" is the mean of the first attribute, and "dev 1" is the standard deviation of the first attribute.

```
public float[] trainData(int classNum){                    //Train data
      float m1, m2, m3, m4, dev1, dev2, dev3, dev4;        //Define mean_1, mean_2…,
std_deviation_1…
      NormalDistributionEstimator est1, est2, est3, est4;
      est1 = new NormalDistributionEstimator();            //Create normal distribution estimator
      est2 = new NormalDistributionEstimator();
      est3 = new NormalDistributionEstimator();
      est4 = new NormalDistributionEstimator();
      float[] accelGroup = new float[4];
      for (int i = 0; i < trainSamplesNum; i++) {
         try {
            float accelX = 0f, accelY = 0f, accelZ = 0f, accelS = 0f;
            for (int j = 0; j < windowSampleNum; j++) {          // windowSampleNum=4
               accelX = accelX + roundDouble(accel.getAccelX() * ENLARGE);   //Attribute 1
               accelY = accelY + roundDouble(accel.getAccelY() * ENLARGE);   //Attribute 2
               accelZ = accelZ + roundDouble(accel.getAccelZ() * ENLARGE);   //Attribute 3
               accelS = accelS + roundDouble(this.get_subData() * ENLARGE);
                                                          //Attribute 4, from slave sensor
               Utils.sleep(1000 / SAMPLE_RATE);           // Set sample rate,
SAMPLE_RATE=20Hz
            }
            accelX = accelX / windowSampleNum;           //Calculate mean
            accelY = accelY / windowSampleNum;
            accelZ = accelZ / windowSampleNum;
            accelS = accelS / windowSampleNum;
            accelGroup[0] = accelX;
            accelGroup[1] = accelY;
            accelGroup[2] = accelZ;
            accelGroup[3] = accelS;
            sendWindowData(accelGroup, classNum);        // Send train data to host
            est1.addValue(accelX, 1.0f);                 //Update normal distribution estimator 1
            est2.addValue(accelY, 1.0f);                 //Update normal distribution estimator 2
            est3.addValue(accelZ, 1.0f);                 //Update normal distribution estimator 3
            est4.addValue(accelS, 1.0f);                 //Update normal distribution estimator 4
         } catch (IOException ex) {
            ex.printStackTrace();
         }
      }
```

Figure 5.5 The code to train data

After training, some important parameters are extracted from raw acceleration data. Table 5-1 shows the parameters of each pattern to build the Gaussian model. Since the floating point operation in ARM9 microprocessor is not as efficient as integer arithmetic operation, the raw acceleration data are enlarged by 100 times to simplify the calculation, as shown in Table 5-1.

Table 5-1 Parameters of each pattern to build Gaussian model

		A1 (g*0.01)	**A2 (g*0.01)**	**A3 (g*0.01)**	**A4 (g*0.01)**
Sitting	Mean	-7.98201	-32.9249	100.3155	95.341
	Standard Deviation	4.757196	3.285977	1.896776	2.624813
Standing	Mean	-0.36251	-104.837	-26.3729	110.7419
	Standard Deviation	4.598349	1.632812	6.829213	0.61885
Lying	Mean	-20.1195	-9.62339	101.0137	93.17715
	Standard Deviation	14.68993	3.86268	2.937374	1.649006
Walking	Mean	-5.93784	-113.608	-5.10875	131.3549
	Standard Deviation	16.37643	27.48725	26.119	25.0675
Walking up the stairs	Mean	-15.9607	-109.788	13.71509	114.0934
	Standard Deviation	11.06533	34.63792	22.49708	26.40725

Based on the data in Table 5-1, we visualize the Gaussian distribution models of each pattern, as shown in Figure 5.6 to Figure 5.9. In Figure 5.6, the horizontal axis represents the acceleration data in X-axis (master sensor) before normalization, and the vertical axis represents the probability density. Five different colors are used to denote five patterns, similarly to Figure 5.7, Figure 5.8 and Figure 5.9. From the four figures, five patterns can be differentiated due to the obvious peaks.

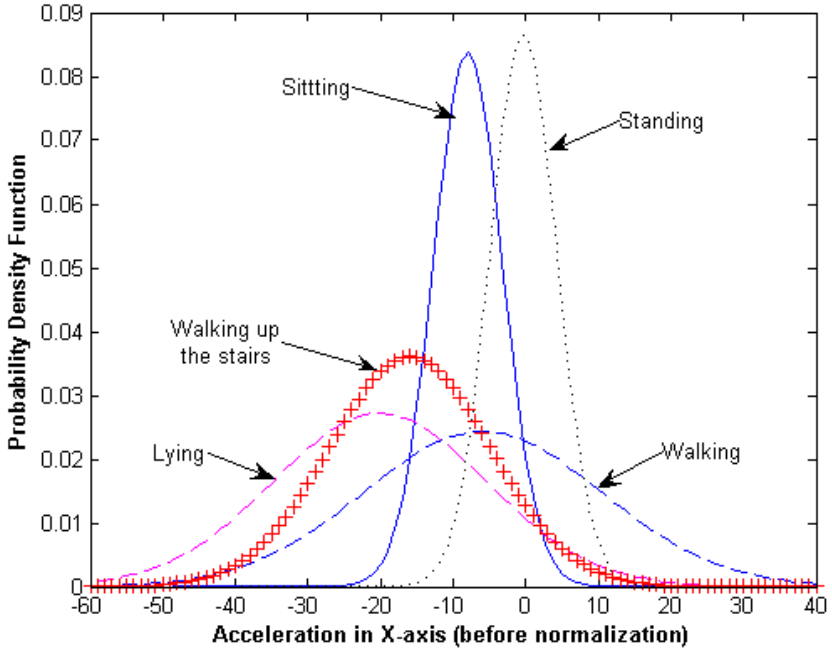

Figure 5.6 Gaussian distribution models of attribute 1, acceleration in X-axis

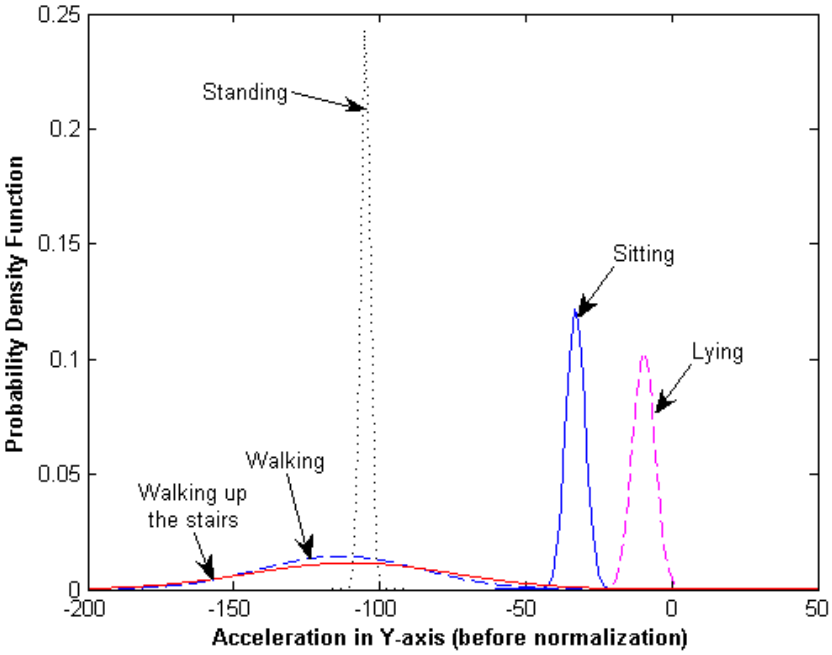

Figure 5.7 Gaussian distribution models of attribute 2, acceleration in Y-axis

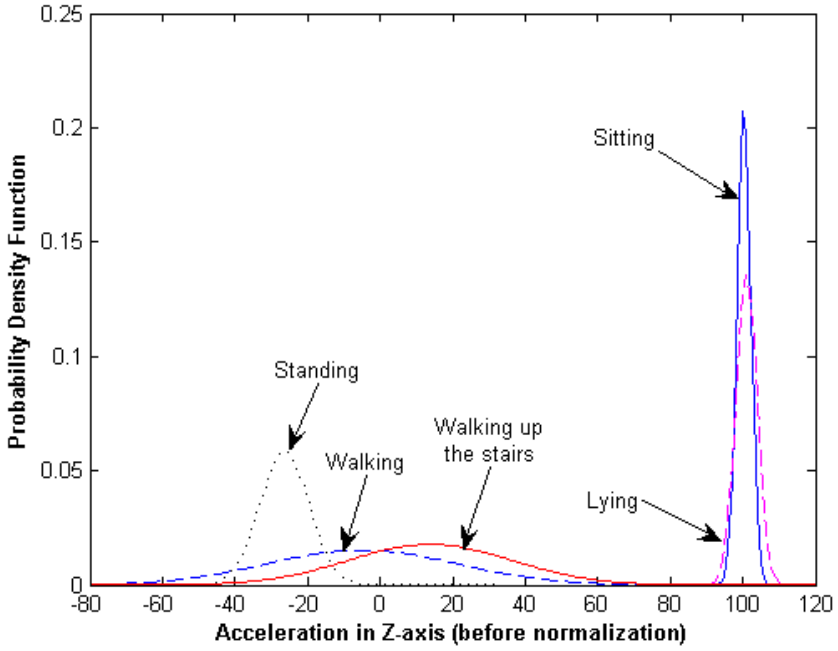

Figure 5.8 Gaussian distribution models of attribute 3, acceleration in Z-axis

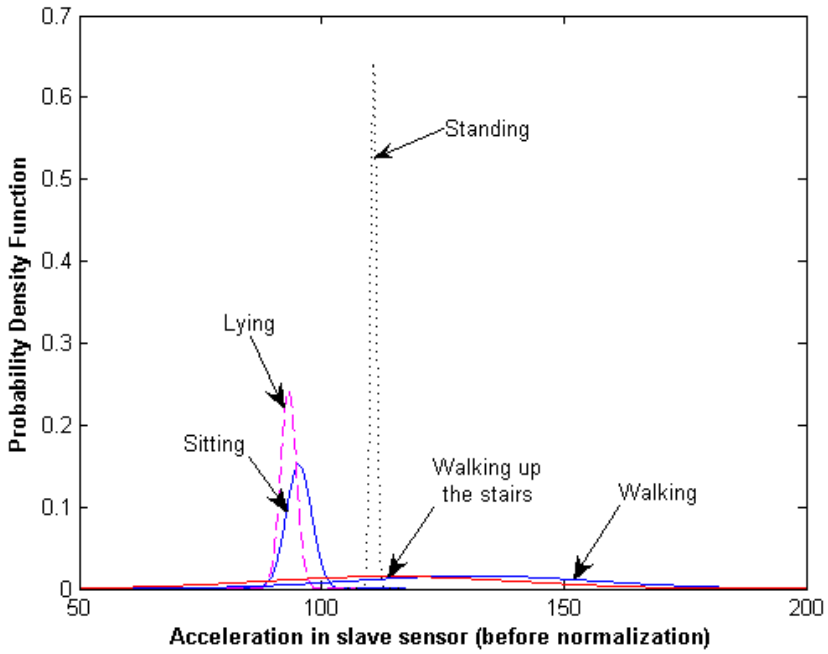

Figure 5.9 Gaussian distribution models of attribute 4, acceleration in the slave sensor

In terms of fall detection, we make some modification in the training process to improve the detection accuracy. As mentioned above, mean and standard deviation are two important parameters in the Gaussian model. Falls can be distinguished from normal physical activity due to their characteristics, according to Figure 4.2 and Figure 4.3 in Chapter 4, e.g., forward fall has a negative peak in Z-axis acceleration; backward fall has a positive peak in Z-axis acceleration, leftward fall has a positive peak in X-axis acceleration; rightward fall has a negative peak in X-axis acceleration. Based on these characteristics, in the training process, each type of fall is trained three times, and the mean of the Gaussian model is replaced by half the average peak value of three falls. This modification makes the classifier detect different falls more accurately. The different fall patterns can also generate similar Gaussian models.

2) Testing:

After training, switch 2 (i.e., pushbutton 2) is activated to start testing process. Multicolor LEDs are also used to denote testing status. Raw acceleration data are firstly divided into small windows, and the mean of each window is calculated. Then the mean is imported to the Gaussian models to calculate probability.

Recall the principle of Naive Bayes in Chapter 4, we have the following equations.

$$p(\text{sit}|A1, A2, A3, A4) = p(\text{sit}) \cdot p(A_1|\text{sit}) \cdot p(A_2|\text{sit}) \cdot p(A_3|\text{sit}) \cdot p(A_4|\text{sit}) \quad (5.1)$$

$$p(\text{stand}|A1, A2, A3, A4) = p(\text{stand}) \cdot p(A_1|\text{stand}) \cdot p(A_2|\text{stand}) \cdot p(A_3|\text{stand}) \cdot p(A_4|\text{stand}) \quad (5.2)$$

$$p(\text{lie}|A1, A2, A3, A4) = p(\text{lie}) \cdot p(A_1|\text{lie}) \cdot p(A_2|\text{lie}) \cdot p(A_3|\text{lie}) \cdot p(A_4|\text{lie}) \quad (5.3)$$

$$p(\text{walk}|A1, A2, A3, A4) = p(\text{walk}) \cdot p(A_1|\text{walk}) \cdot p(A_2|\text{walk}) \cdot p(A_3|\text{walk}) p(A_4|\text{walk}) \quad (5.4)$$

$$p\left(\text{walk}_{\text{up}}|A1, A2, A3, A4\right) = p\left(\text{walk}_{\text{up}}\right) \cdot p(A_1|\text{walk}_{\text{up}}) \cdot p(A_2|\text{walk}_{\text{up}}) \cdot p(A_3|\text{walk}_{\text{up}}) \cdot p(A_4|\text{walk}_{\text{up}}) \quad (5.5)$$

where p(sit), p(stand), p(lie), p(walk) and $p(\text{walk}_{\text{up}})$ are called prior probabilities. Since the training data of each pattern are sampled with the same number, their prior probabilities are equal to a constant value 1/5. Since there are five patterns or classes,

each unknown testing data will produce five probabilities. The pattern with highest probability is considered as the classification result. Code in Figure 5.10 is for testing and generating classification results; each operation process consumes about 200ms.

```
public int testData(float[] guassianParaPattern1, float[] guassianParaPattern2, float[] guassianParaPattern3, float[] guassianParaPattern4, float[] guassianParaPattern5) {
    int result = 0;
    float[] data = new float[4];
    try {
      float accelX = 0f, accelY = 0f, accelZ = 0f, accelS = 0;
      for (int j = 0; j < windowSampleNum; j++) {
        accelX = accelX + roundDouble(accel.getAccelX() * ENLARGE);
        accelY = accelY + roundDouble(accel.getAccelY() * ENLARGE);
        accelZ = accelZ + roundDouble(accel.getAccelZ() * ENLARGE);
        accelS = accelS + roundDouble(this.get_subData() * ENLARGE);
        Utils.sleep(1000 / SAMPLE_RATE);                    // Set sample rate
      }
      accelX = accelX / windowSampleNum;                    //Calculate mean
      accelY = accelY / windowSampleNum;
      accelZ = accelZ / windowSampleNum;
      accelS = accelS / windowSampleNum;
      data[0] = accelX;
      data[1] = accelY;
      data[2] = accelZ;
      data[3] = accelS;
      double[] pPattern = new double[5];
      pPattern[0] = calculateGuassian(guassianParaPattern1, data);      //Calculate probability
      pPattern[1] = calculateGuassian(guassianParaPattern2, data);
      pPattern[2] = calculateGuassian(guassianParaPattern3, data);
      pPattern[3] = calculateGuassian(guassianParaPattern4, data);
      pPattern[4] = calculateGuassian(guassianParaPattern5, data);
      double maxProbability = 0;
      for (int i = 0; i < 5; i++) {                 // Find the max probability
        if (maxProbability < pPattern[i]){
          maxProbability = pPattern[i];
          result = i+1;                             // Make the decision based on max probability
        }
      }
      sendWindowData(data, result);   //Send the testing data and classification result to host
    } catch (IOException ex) {
      ex.printStackTrace();
    }
    return result;
  }
```

Figure 5.10 The code for testing and generating classification results; each operation process takes about 200ms

5.2 Implementation on laptop

According to Figure 5.1 the block diagram of the system set-up, although all the data collection and data processing are accomplished on Sun SPOT sensor, a user-friendly graphic user interface (GUI) on laptop will provide users more details. GUI can display raw acceleration data of all axes in real time. In indoor application, users can monitor the movement of themselves, as shown in Figure 5.11 to Figure 5.13. The acceleration data of five physical activities are displayed. These data are collected when the sensor is attached to the thigh of the testing subject. Horizontal axis is time; vertical axis represents acceleration (unit g). Green curve is the acceleration data in X-axis; Blue curve is the acceleration data in Y-axis; Red curve is the acceleration in Z-axis.

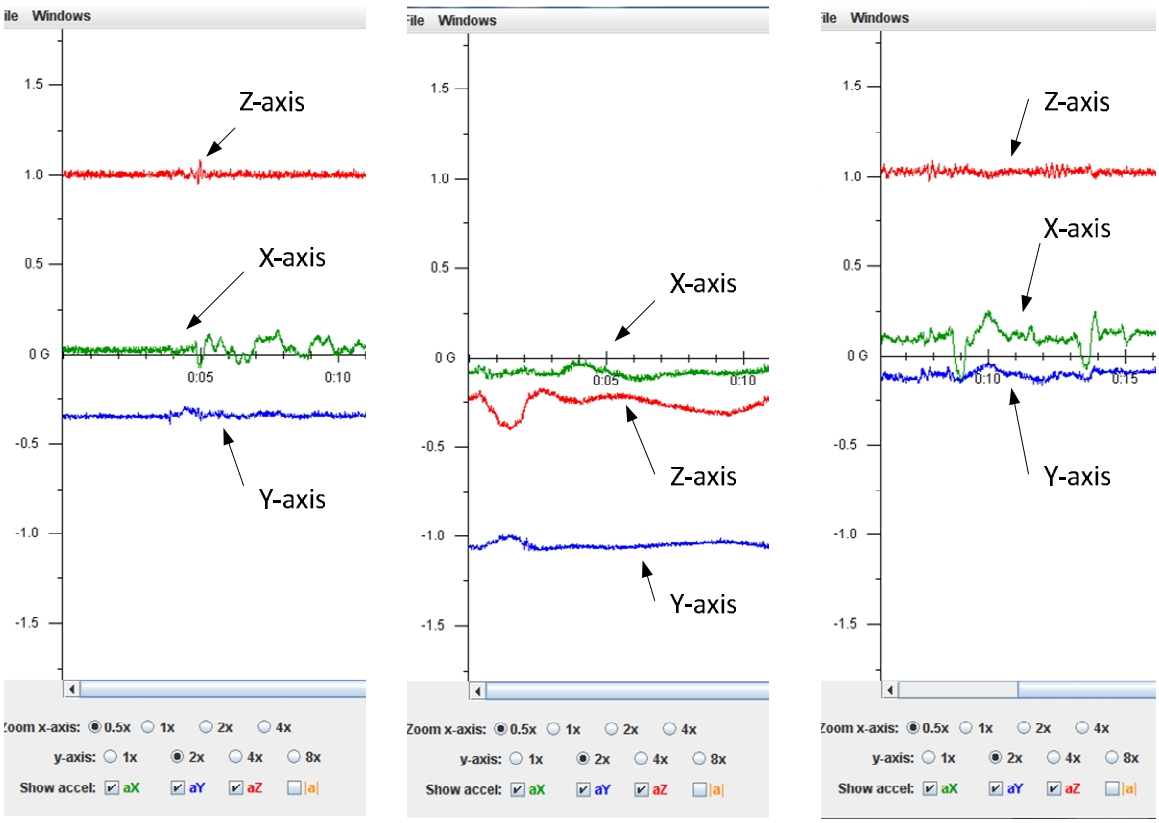

Figure 5.11 GUI displays physical activity: sitting (left), standing (middle), and lying (right)

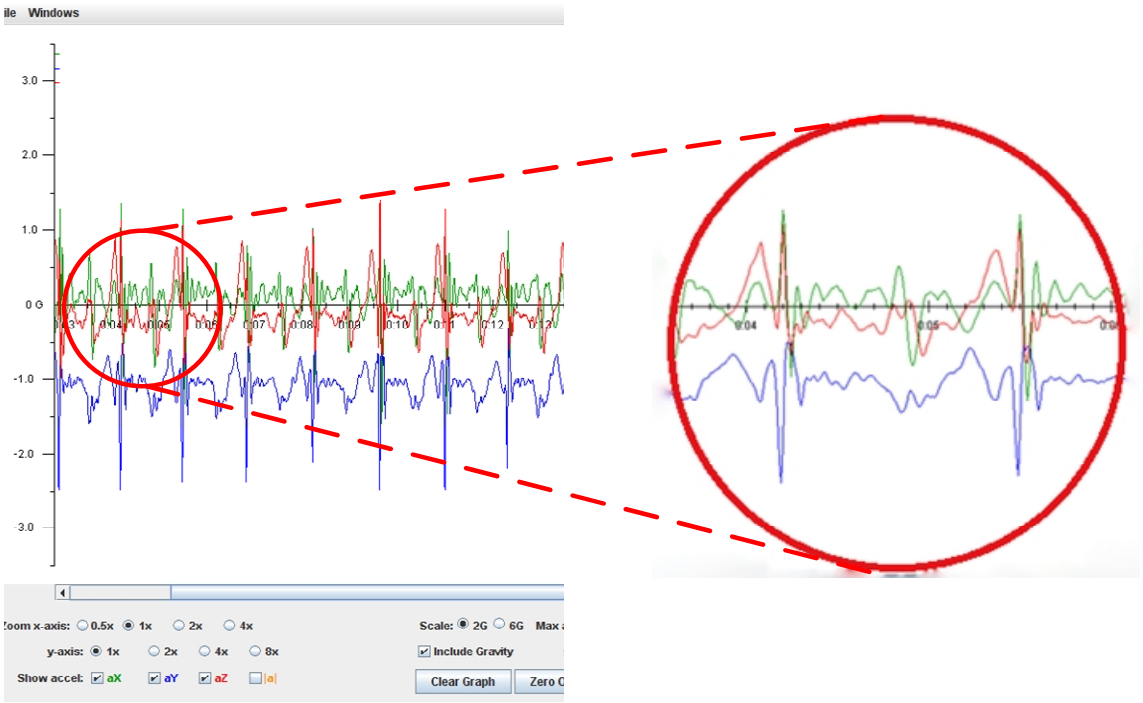

Figure 5.12 GUI displays physical activity: walking. Left picture shows the original display; right picture is enlarged by 4 times in the horizontal axis

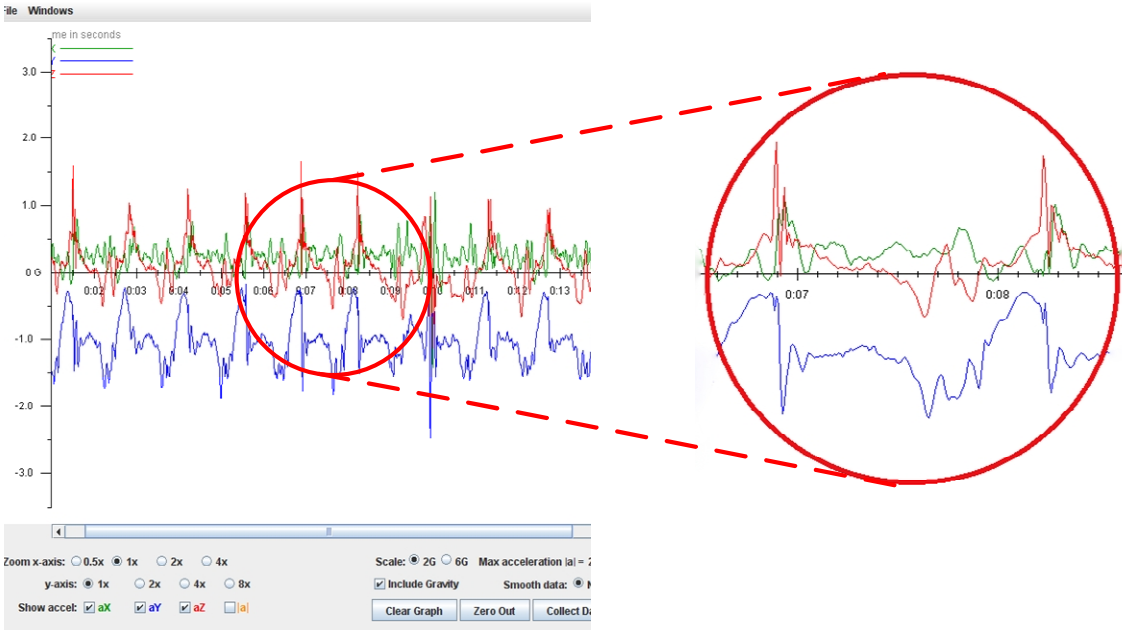

Figure 5.13 GUI displays physical activity: walking up the stairs. Left picture shows the original display; right picture is enlarged by 4 times in the horizontal axis

Once the raw data are received in the laptop, extensive computation on the data can be performed in the laptop, such as FFT, wavelet transform. In addition, the memory space in each Sun SPOT sensor is limited (512K RAM and 4M Flash), which cannot satisfy long term monitoring. Thus we store the receiving data in .CSV file, and users can access it anytime. Figure 5.14 shows the acceleration data are stored before normalization. From column A to column D are data, and the last column shows the class index. Class 1 means sitting, class 2 means standing, etc.

	A	B	C	D	E
1	-7.78733	-33.5661	101.2352	96.44933	1
2	-8.59291	-33.2975	100.6982	94.88471	1
3	-10.8754	-33.029	99.48979	95.39471	1
4	…	…	…	…	…
5	-5.37057	-108.888	-3.35661	109.6242	2
6	-4.96778	-109.291	-7.116	109.6113	2
7	-2.95381	-108.083	-13.5607	109.7432	2
8	…	…	…	…	…
9	-22.9592	-9.66703	100.5639	93.13551	3
10	-19.3341	-10.4726	103.652	93.1715	3
11	-14.3663	-13.0236	101.638	92.4584	3
12	…	…	…	…	…
13	-6.31042	-85.6606	20.81096	111.386	4
14	13.56069	-88.8829	5.504833	163.9184	4
15	-22.8249	-99.8926	-12.3523	106.1679	4
16	…	…	…	…	…
17	-12.4866	-105.666	-4.02793	99.972	5
18	-9.93555	-139.232	21.34801	107.9809	5
19	12.7551	-93.985	-22.6907	109.1115	5
20					

Figure 5.14 Acceleration data are recorded in a ".CSV" file

Chapter 6 Results and Discussion

This chapter includes four parts: (1) Evaluate sampling rate when recognizing physical activity. Accuracy comparison at different sampling rates on the Sun SPOT is discussed and an optimal sampling rate is selected. (2) Evaluate physical activity recognition when one sensor is used. The accuracy comparison is discussed when attaching only one sensor at different location on the human body. The location which provides the most useful information is determined. (3) Evaluate physical activity recognition when two sensors are used. (4) Evaluate fall detection.

All the sampling data are collected from the same subject, with each activity trained 160 times and tested 400 times.

6.1 Sampling rate evaluation

Previous research mentioned the bandwidth of the characteristic mobility data is less than 3 Hz [33]. Based on the Nyquist Sampling Theorem, the sampling rate should be at least two times the original signal frequency without losing useful information. Provided that the signal is sampled at an extremely high sampling rate, the sampled signal is extremely close to the actual signal, and fewer details are lost. However, high sampling rate may cause redundant data, which will slow down the processing speed.

In this test environment, only one sensor was attached to the right thigh of the subject and 5 physical patterns were trained and tested. There are 40 training sets as well as 100

testing sets for each pattern. Each segment was a 0.2 second time-domain window and each segment produced a classification result.

Table 6-1 shows the testing result. The data were sampled at a frequency from 5Hz to 40Hz. Five patterns were performed. The accuracy of each pattern at a particular sampling rate was calculated. Then the overall accuracy was also calculated. The accuracy increases with the increasing sampling rate as shown in Figure 6.1. The accuracy ranges from 84% to 87.8%. However, beyond 20Hz, there is not much gain in accuracy. Therefore, for this particular testing environment, 20Hz is selected as the optimal sampling rate.

Table 6-1 Accuracy (%) comparison with different sampling rates

Sampling rate	5Hz	10Hz	20Hz	30Hz	40Hz
Sitting	100	100	100	100	100
Standing	100	100	100	100	100
Lying	100	100	100	100	100
Walking	36.1	60	53.13	57.14	61.7
Walking up	80	63.27	82.65	77	74.23
Overall	84	84.8	87.55	87	87.8

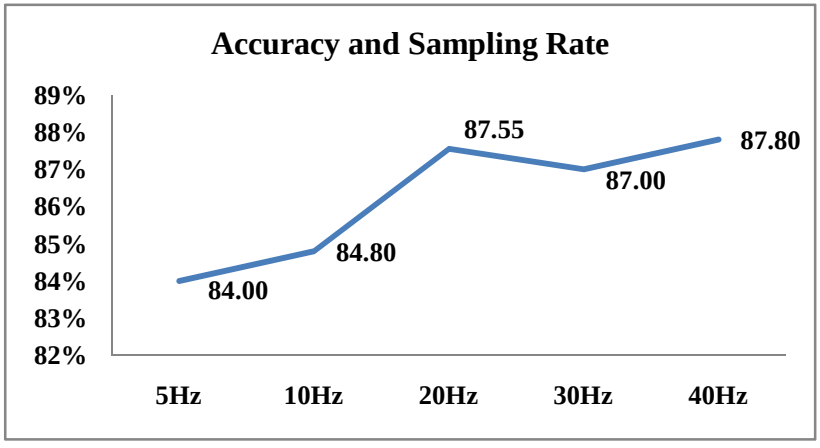

Figure 6.1 Accuracy (%) variance with different sampling rates

6.2 Evaluation of physical activity recognition with one sensor

As mentioned in the previous chapter, different sensor placement on a human body will provide different movement information. For example, the sensor attached at the thigh provides more information than at head when the testing subject is walking. Experiments were performed to evaluate the physical activity recognition system.

At the 20Hz sampling rate, one sensor was attached to different locations: thigh, calf, and chest. Acceleration in x, y, and z axes provide three attributes for the Naive Bayes classifier. Figure 6.2 shows the accuracy variance when one sensor is attached to different locations. Physical activities are displayed; A: sitting; B: standing; C: lying; D: walking; E: walking up the stairs. The first six bars represent the recognition accuracy when the sensor is at the thigh.

From Figure 6.2, when one sensor is attached to the thigh, it provides the highest overall accuracy, 87.55%. Static physical activities are recognized accurately. When one sensor is at the calf, this location has the lowest overall accuracy, only 69.22%. Less than 50% sitting and walking up the stairs can be recognized successfully. When one sensor is attached to the chest, standing is often misclassified as other patterns because a very light sway may cause the misclassification. As a result, if using only one sensor to recognize physical activity, the best location is attaching it to the thigh.

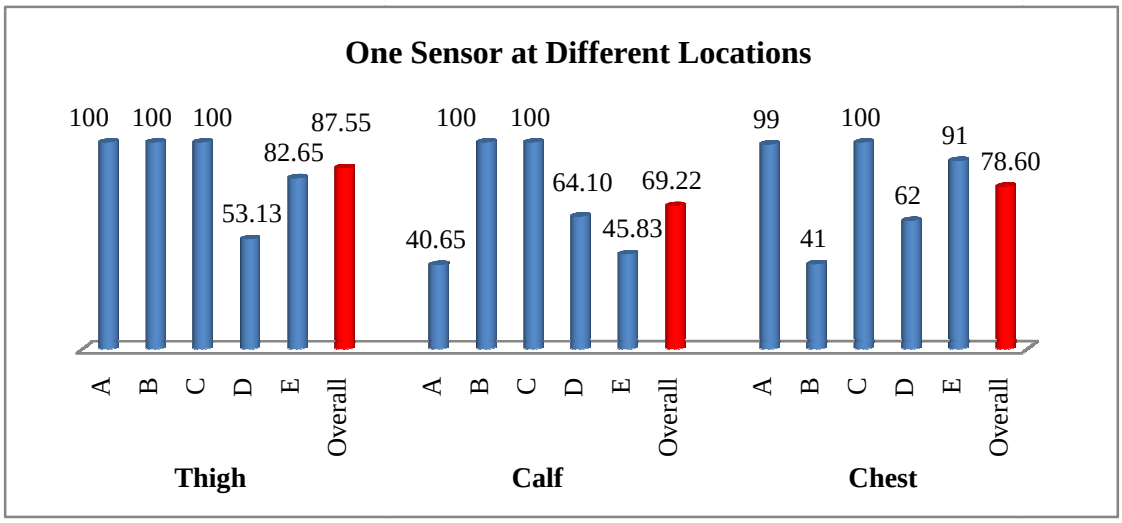

Figure 6.2 Different accuracy (%) with one sensor at different locations: thigh, calf and chest. A: sitting; B: standing; C: lying; D: walking; E: walking up the stairs

In order to evaluate the performance of a classifier, confusion matrix is used to indicate the results. In a confusion matrix, the horizontal direction denotes that the pattern actually happens, while the vertical direction denotes the "classified as" patterns. For example, there are 96 samples in D pattern, but 14 of them are misclassified as B, 31 of them are misclassified as E, only 51 samples are correctly classified.

Table 6-2 is the confusion matrix of the classification results when one sensor is attached to the thigh. From the table, static physical activities, such as sitting, standing or lying, are accurately recognized. But 46.8% walking is misclassified as walking up the stairs or standing; 17% walking up the stairs is misclassified as walking.

Table 6-2 Confusion matrix of the classification results when one sensor is attached to the thigh

Physical Activity		**Classified As**				
		A	B	C	D	E
A:Sitting B:Standing C:Lying D:Walking E:Walking up the stairs	A	104	0	0	0	0
	B	0	100	0	0	0
	C	**0**	0	100	0	0
	D	0	14	0	51	31
	E	0	1	0	16	81

6.3 Evaluation of physical activity recognition with two sensors

At the 20Hz sampling rate, two sensors were attached to different locations: right thigh & right calf, right thigh & chest, and left thigh & right thigh. The master sensor was always attached to the right thigh, and the slave sensor changed locations. Acceleration in x, y, and z axes in the master sensor provided three attributes for the Naive Bayes classifier, and the slave sensor provided the fourth attribute. Figure 6.3 shows the accuracy variance when two sensors are attached to different locations. In Figure 6.3, different physical activities are displayed; A: sitting; B: standing; C: lying; D: walking; E: walking up the stairs.

For overall accuracy, Figure 6.3 indicates that the combination of left thigh and right thigh has the highest accuracy than the others. No matter what the combination is, the system with two sensors achieves higher accuracy than one sensor. Different combination of two sensors shows little variance in accuracy (89.78%, 89.80% and 90.52%). Thus, if two sensors are available, for best accuracy, attaching them to the left thigh and the right thigh is the solution.

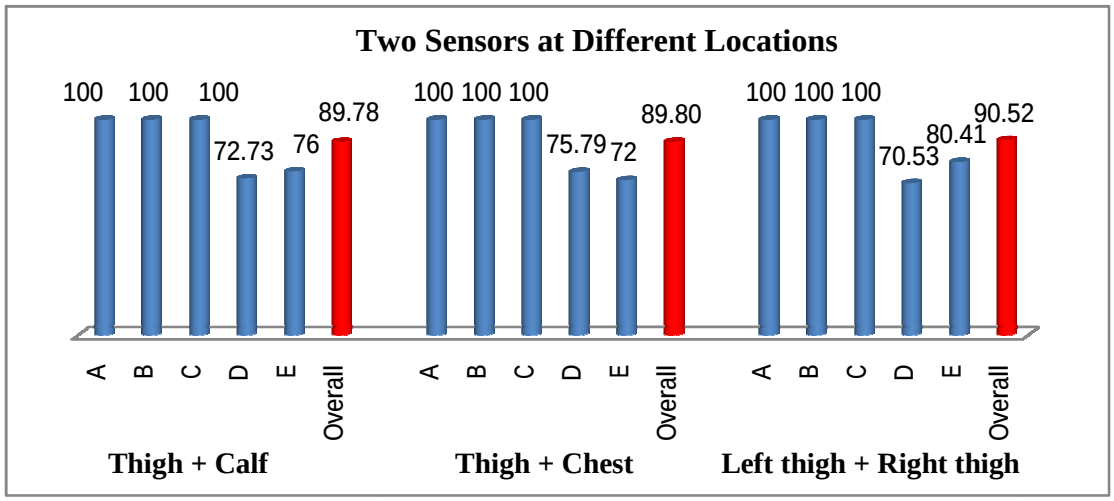

Figure 6.3 Different accuracy (%) with two sensors at different locations: thigh + calf, thigh + chest, left thigh + right thigh. A: sitting; B: standing; C: lying; D: walking; E: walking up the stairs

The confusion matrix in Table 6-3 shows the classification results when two sensors are attached to the left and the right thigh, where 29% walking is misclassified to walking up the stairs and 19% walking up the stairs is misclassified to walking. Static physical activity is classified accurately.

Table 6-3 Confusion matrix of the classification results when two sensors are attached to the left thigh and to the right thigh

Physical Activity		**Classified As**				
		A	B	C	D	E
A:Sitting B:Standing C:Lying D:Walking E:Walking up the stairs	A	108	0	0	0	0
	B	0	98	0	0	0
	C	0	0	98	0	0
	D	0	0	0	67	28
	E	0	0	0	19	78

6.4 Evaluation of fall detection with two sensors

In fall detection, the master sensor is attached to the chest because the sensor at the chest can provide more movement information than other parts when the subject is falling down. The slave sensor is attached to the right thigh. In Figure 6.4, a testing subject wearing two sensors falls down from a static standing position to a 34 cm height mattress.

Figure 6.4 Fall detection environment

Recalling Figure 1.3 in Chapter 1, the timing of falling down is illustrated. The actual testing experiments demonstrate that the calculation time of fall detection system (i.e., fall detection time) is about 200ms. The calculation time plus the mechanical response time of the protection system is about 300ms, which is much less than falling time (2000ms). That means with current technique a fall can be successfully prevented.

In this application, falls are supposed to be detected from static standing and walking. There are 6 recognition patterns: forward fall, backward fall, leftward fall, rightward fall, standing and walking. In the training process, each type of fall is serially performed 3 times (4 types of fall * 3 times = 12 times); standing and walking are performed 8 seconds respectively. In the testing process, each type of fall is performed about 20 times, totally about 80 times; standing and walking are performed 20 seconds respectively.

Fall detection is considered to be positive if the detector properly recognizes a fall, to be negative if it does not. So there are four cases:

1) True positive (TP): a fall occurs, the device detects it.

2) False positive (FP): the device announces a fall, but it did not occur.

3) True negative (TN): a normal (no fall) movement is performed, the device does not declare a fall.

4) False negative (FN): a fall occurs but the device does not detect it.

In order to evaluate the performance of different fall detection systems, Noury et al. [34] proposed a criterion, called sensitivity.

$$Sensitivity = \frac{TP}{TP + FN} \tag{6.1}$$

It is used to evaluate the capacity to detect a fall. Higher sensitivity indicates that the fall detection system could detect falls accurately with less false alarms.

Table 6-4 shows the testing results, a confusion matrix of the fall detection results. Different activities are represented by A to F. Totally 88 falls are performed, and each fall is successfully detected as fall; although one particular fall is possible to be classified as another fall (e.g., forward fall is classified as leftward fall once and backward fall once, as shown in Table 6-4). Normal activities (i.e., standing or walking) are impossible to be recognized as falls, indicating there is no false alarm while the user is wearing this system in daily life.

Table 6-4 Confusion matrix of the fall detection results

Activity		**Classified As**					
		A	B	C	D	E	F
A: Forward fall B: Backward fall C: Leftward fall D: Rightward fall E: Standing F: Walking	A	22	1	1	0	0	0
	B	1	18	1	2	0	0
	C	1	0	19	0	0	0
	D	2	1	1	18	0	0
	E	0	0	0	0	100	0
	F	0	0	0	0	2	98

Table 6-5 shows the statistic of the fall detection results. In the second column, 24 forward falls are performed, and the fall detection system detects 22 forward falls, 1 backward fall, and 1 leftward fall. Thus the sensitivity is 22/24, 92%. Backward fall is 82% sensitivity; leftward fall is 95% sensitivity; rightward fall is 82% sensitivity. Seventy seven falls are exactly detected out of 88 falls, so the overall sensitivity is 87.5%.

Table 6-5 Statistic of the fall detection results

Fall	Forward	Backward	Leftward	Rightward	Overall
Exactly detect/Fall	22/24	18/22	19/20	18/22	77/88
Sensitivity %	92	82	95	82	87.5

6.5 Comparisons

The thesis work is compared with a previous thesis work [35] in different aspects.

Table 6-6 Thesis work comparisons

	Previous thesis work [35]	My thesis work
Hardware	Wearable device [36] (tri-axis accelerometer, gyroscope, heart beat sensing circuit, RF-ready microcontroller)	Sun SPOT wireless sensor module (tri-axis accelerometer, ARM9 microprocessor)
Sensor numbers	Single; double	Single; double
Sensor locations	One sensor at chest, second sensor at thigh	One sensor: thigh; chest; calf Two sensors: (1)left thigh + right thigh; (2)right thigh + chest; (3)right thigh + right calf
Data preprocessing (window technique)	Use instant data samples as the input signals to classifier	Preprocess raw data (form small windows), apply algorithm on each window
Sampling rate	40 Hz	5Hz, 10Hz, 20Hz, 30Hz, 40Hz
Accuracy	Single node: 99.403% Double nodes: 99.7817%	(1)Physical activity recognition: One sensor: 87.55% Two sensors: 90.52% (2)Fall detection: Two sensors:87.55% sensitivity
Real-time analysis	No	Yes
Graphic user interface	No	Yes
Timing analysis	No	Yes

Chapter 7 Conclusion and Future Work

7.1 Conclusion

In order to recognize physical activity and detect fall, the characteristics of normal physical activity and fall were studied, based on which, a wearable real-time system is designed. The hardware platform is the Sun SPOT sensor with tri-axis accelerometer, which is used to collect acceleration data. In terms of algorithm, some popular artificial intelligence algorithms were evaluated and compared, including ZeroR, Support Vector Machine (SVM), OneR, C4.5, Neural Network and Naive Bayes, rather than traditional threshold approaches. Offline data processing is performed in WEKA. The evaluation results indicate that Naive Bayes algorithm works better than other popular algorithms both in accuracy and implementation in this particular application.

The real-time system is designed to work in two modes. Mode 1 is particularly for indoor applications, where a GUI provides more data details in the computer used as a monitor. User can monitor the movement of a testing subject and record related data in a designated folder for further processing. Mode 2 is primarily designed for outdoor applications, because it only displays the classification results with multicolor LEDs on the Sun SPOT, providing users more flexibility.

We successfully implemented the Naive Bayes classifier in the Sun SPOT sensor. In order to evaluate the sampling rate, acceleration data are collected while testing subject performs normal physical activity. As the sampling rate increases, from 5 Hz to 40 Hz, there is not much gain in accuracy for the sampling rate higher than 20 Hz. Therefore, for this particular testing environment, 20Hz is selected as the optimal sampling rate. But for

other subjects in different age and different activity patterns, the sampling rate should be further evaluated.

Normal physical activities include sitting, standing, lying, walking and walking up the stairs. If only one sensor is available to recognize physical activity, the best location is attaching it to the thigh as this provides a maximum of 87% overall accuracy. If two sensors are available, the combination of the left thigh and the right thigh is the best option. This combination gives a maximum 90% of overall accuracy. The system with two sensors performs better than that with only one sensor in terms of accuracy.

In fall detection, forward, backward, leftward and rightward falls have been detected. We attached one sensor to the chest, another sensor to the thigh to collect acceleration data. The results showed that all falls were successfully detected; although one particular fall was possible to be classified as another fall. Normal physical activity is not misclassified as fall, indicating there is no false alarm for the use of this system in daily life to detect and prevent fall. The overall sensitivity of the fall detection is about 87%. The time for the system to detect a fall is about 200ms in actual testing experiments, which is much less than the fall time of a subject which is about 2000ms. This fast detection provides plenty of time for an effective fall prevention device to activate.

Although a real-time wearable system is successfully designed and implemented for physical activity recognition as well as fall detection, there are still limitations. All the experiment data were collected from a young healthy subject, and all the classification evaluations were based on those data. In order to find the best solution for a wide range of population with different activity styles, more data should be collected from subjects in various ages.

7.2 Future work

Using acceleration data collected from the Sun SPOT sensors attached to a human body, people's daily activity and energy expenditure (EE) can be estimated and monitored for long term health problem investigation. In many metabolic disorders, measuring daily energy expenditure is necessary [37]. The current gold standard for

measuring energy expenditure is the doubly labeled water (DLW), which provides accurate measurement results. Unfortunately, DLW is rarely used for large populations due to its high cost. Compared to DLW, researchers have developed prediction equation to estimate energy expenditure based on accelerometry and subject characteristics. Kong Y. Chen and Ming Sun [38] have estimated energy expenditure by combining acceleration of all three axes in both linear and nonlinear models. For example, in the linear estimation model, physical activity energy expenditure (EE_{ack}) is estimated by

$$EE_{ack}(k) = a_L \times H(k) + b_L \times V(k) \tag{7.1}$$

where a_L and b_L represent the regression parameters in the linear equation; H(k) is the square root of the sum of squared signals of the X-axis and Y-axis (i.e., $\sqrt{x^2 + y^2}$) at the *k*th minute; V(k) is the acceleration in Z-axis at the *k*th minute. Once the appropriate parameters a_L and b_L have been fixed, equation 7.1 can be used to predict physical activity energy expenditure.

Another future exploration is to use the system to realize telemedicine, as shown in Figure 7.1. Since the Sun SPOT wireless sensors have a built-in HTTP networking capability to interact with a remote server, such as the Twitter service (a social networking and microblogging service). The local data can be transmitted to the remote server; meanwhile the commands or other data can be sent back to the local sensor from the remote sever. In the telemedicine application, local acceleration information and classification results are firstly transmitted to the local laptop wirelessly. The laptop then sends data to the remote server, where the data are stored. Users in remote laptop 1 and remote laptop 2 can log in the remote server to access the stored acceleration information and classification results. Based on this strategy, the medical staffs or the user's relatives and friends at remote terminal can access user's data anytime.

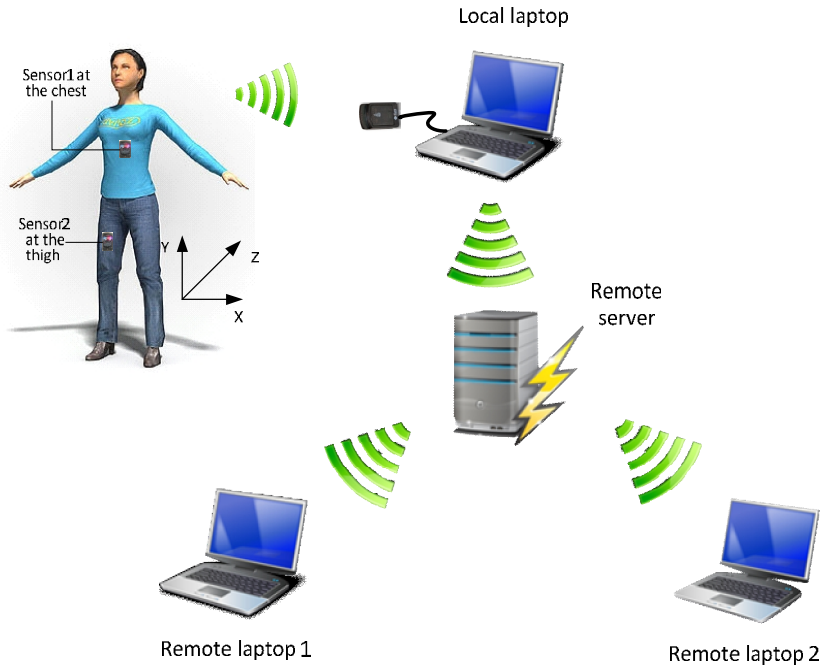

Figure 7.1 Future exploration: telemedicine, remote laptops access the data via a remote server

Bibliography

[1] C. Caspersen, K. Powell, and G. Christenso, "Physical Activity, Exercise, and Physical Fitness: Definitions and Distinctions for Health-Related Research," *Public Health Reports*, vol. 100 , no. 2 , pp. 126-31, 1985.

[2] S. Preece et al., "Activity identification using body-mounted sensors—a review of classification techniques," *Physiological Measurement*, vol. 30, no. 4, pp. R1-R33, April 2009.

[3] E.M. Tapia et al., "Real-Time Recognition of Physical Activities and Their Intensities Using Wireless Accelerometers and a Heart Rate Monitor," in *Wearable Computers, 11th IEEE International*, 2007, pp. 37-40.

[4] Steven T. Moore, Hamish G. MacDougall, and William G. Ondo, "Ambulatory monitoring of freezing of gait in Parkinson's disease," *Journal of Neuroscience Methods* , vol. 167, pp. 340–348, 2008.

[5] Division of Aging and Seniors, "Report on Seniors' Falls in Canada," Public Health Agency of Canada, Ottawa, Ontario, 2010.

[6] Ministry of Health Planning Officer of the Provincial Health Officer, "Prevention of Falls and Injuries Among the Elderly," British Columbia, January 2004.

[7] K. Tabata, "Population aging, the costs of health care for the elderly and growth," *Journal of Macroeconomics*, vol. 27, no. 3, pp. 472-493, September 2005.

[8] Aging and the Canadian population, 2010, http://www.mta.ca/faculty/arts/canadian_studies/english/about/aging/.

[9] Canadian Institute for Health Information, "National trauma registry 2004 report: Injury hospitalizations (includes 2002-2003 data)," Ottawa, 2004.

[10] Canadian Institute for Health Information, "National trauma 2003 registry report: Injury hospitalizations (includes 2001-2002 data)," Ottawa, 2004.

[11] M.E. Tinetti and C.S. Williams, "Falls, injuries due to falls, and the risk of admission to a nursing home," *New England Journal of Medicine*, vol. 337, no. 18, pp. 1279-84, 1997.

[12] R.G. Cumming, G. Salkeld, M. Thomas, and G. Szonyi, "Prospective study of the impact of fear of falling on activities of daily living , SF-36 scores, and nursing

home admission," *Journal of Gerontology*, vol. 55, no. 5, pp. M299-M305, 2000.

[13] Health Canada, "Seniors and aging – preventing falls in and around your home," 2010.

[14] S. R. Lord, C. Sherrington, and H. B. Menz, *"Falls in older people: risk factors and strategies for prevention," Injury prevention*.: Cambridge University Press, 2001.

[15] T. W. O'Neill et al., "Age and sex influences on fall characteristics," *Annual of the Rheumatic Diseases*, pp. 53(11): 773-775, Nov. 1994.

[16] E. T. Hsiao and S. N. Robinovitch, "Common protective movements govern unexpected falls from standing height," *Journal of Biomechanical Engineering*, pp. 31(1):1-9, Jan. 1998.

[17] S.N. Robinovitch, W.C. Hayes, and T.A. McMahon, "Prediction of femoral impact forces in falls on the hip," *Journal of Biomechanical Engineering*, pp. 113(4):366-74, Nov. 1991.

[18] X. Yu, "Approaches and Principles of Fall Detection for Elderly and Patient," in *e-health Networking, Applications and Services. HealthCom 10th International Conference on*, Singapore, 7-9 July 2008, pp. 43-47.

[19] T. Tamura, T. Yoshimura, M. Sekine, M. Uchida, and Osamu Tanaka, "A Wearable Airbag to Prevent Fall Injuries," *IEEE Transactions on Information Technology in Biomedicine*, vol. 13, no. 6, pp. 910-914, November 2009.

[20] G. Plasqui and K. R. Westerterp, "Physical activity assessment with accelerometers: an evaluation against doubly labeled water," *Obesity (Silver Spring)*, vol. 15, no. 10, pp. 2371-9, Oct. 2007.

[21] A. Godfreya, R. Conwaya, D. Meagherd, and G. ÓLaighin, "Direct measurement of human movement by accelerometry," *Medical Engineering and Physics*, vol. 30, no. 10, pp. 1364-1386, 2008.

[22] Sun Microsystems Inc. (2009, Nov.) sunspotworld. [Online]. http://www.sunspotworld.com/docs/Red/Tutorial/Tutorial.html

[23] Sun Microsystems Inc. (2009, Nov.) sunspotworld. [Online]. http://www.sunspotworld.com/GettingStarted/index.html

[24] Ron Goldman. (2007, Feb.) Using the LIS3L02AQ Accelerometer. A Sun SPOT application note.

[25] M. Hall et al., "The Weka data mining software: an update," *SIGKDD Explorations*, vol. 11, no. 1, pp. 10-18, 2009.

[26] Maimon Oded and Lior Rokach, "Chapter 9: Decision Trees," in *Data Mining and Knowledge Discovery Handbook.*: Springer, 2005.

[27] Krzysztof J. Cios, Witold Pedrycz, Roman W. Swiniarski, and Lukasz A. Kurgan, *Data Mining: A Knowledge Discovery Approach*. New York, USA: Springer, 2007.

[28] C. J. C. Burges, "Data Mining and Knowledge Discovery," *A tutorial on support vector machines for pattern recognition*, vol. 2, pp. 121-167, 1998.

[29] L. Auria and R.A. Moro, *Support Vector Machines (SVM) as a Technique for Solvency Analysis*. DIW Berlin, Germany: German Institute for Economic Research, August 2008.

[30] Nicolas Galoppo von Borries. Introduction to Artificial Neural Networks. Lecture slide, COMP290-058 Motion Planning.

[31] Torsten Reil. Artificial Neural Networks. Lecture slide.

[32] H. Zhang and J. Su, "Naive Bayes for optimal ranking," *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 20, pp. 79-93, 2008.

[33] G. M. Lyons, K. M. Culhane, D. Hilton, P. A. Grace, and D. Lyons, "A description of an accelerometer-based mobility monitoring technique," *Medical Engineering & Physics*, vol. 27, no. 6, pp. 497-504, July 2005.

[34] N. Noury et al., "Fall detection - principles and methods," in *Proceedings of the 29th annual international conference of the IEEE EMBS*, France, August 23-26,2007, pp. 1663-1666.

[35] A. Ralhan, *Machine learning for Fall Detection and Movement Classification*, M.Sc. thesis, Ed. Saskatoon, Canada: University of Saskatchewan, Dec. 2009.

[36] A. Dinh et al., "Implementation of a Physical Activity Monitoring System for the Elderly People with Built-in Vital Sign and Fall Detection," in *Sixth International Conference on Information Technology: New Generations*, Las Vegas, Nevada, USA, 2009, pp. 1226-1231.

[37] A. G. Bonomi, G. Plasqui, A. H. C. Goris, and K. R. Westerterp, "Improving assessment of daily energy expenditure by identifying types of physical activity with a single accelerometer," *Journal of Applied Physiology*, pp. 655-61, June 25 2009.

[38] Kong Y. Chen and Ming Sun, "Improving energy expenditure estimation by using a triaxial accelerometer," *Journal of Applied Physiology*, vol. 83, no. 6, pp. 2112-2122, December 1997.